NEUE PARADIGMEN IN DER DEZENTRALEN PROZESSKONTROLLDATENVERWALTUNG: VON BLOCKCHAIN-SYSTEMEN BIS ZU TRADITIONELLEN SYNCHRONISIERUNGSALGORITHMEN

Von der Universität Bayreuth
zur Erlangung des Grades eines
Doktors der Naturwissenschaften (Dr. rer. nat.)
genehmigte Abhandlung

von
Christian Sturm
aus Neustadt a.d. Waldnaab

1. Gutachter: Prof. Dr.-Ing. Stefan Jablonski
2. Gutachter: Prof. Dr. Gilbert Fridgen

Tag der Einreichung: 06. Juni 2024
Tag des Kolloquiums: 18. Juli 2024

UNIVERSITÄT
BAYREUTH

Für meine Familie.

Tanja

Helmut
Angelika
Stefan
Melanie
Rudolf†
Bernd
Natali
Anika
Martin
Marie
Lore

ZUSAMMENFASSUNG

Der Forschungsbereich des Geschäftsprozessmanagements befasst sich unter anderem mit der Modellierung von Geschäftsprozessen, der Implementierung und Ausführung der Prozessmodelle in Ausführungssystemen sowie der Analyse und Optimierung von Prozessen. Dabei beschreiben Geschäftsprozesse wiederholt ablaufende Vorgänge in Unternehmen. Über die softwaregestützte Ausführung werden die Prozessmodelle und die darin beschriebenen Teilschritte aktiv in Prozessausführungssystemen (englisch: *workflow management system*, WFMS) interpretiert. Darüber lassen sich zum Beispiel zu erledigende Teilschritte über alle Mitarbeitenden hinweg koordinieren, indem die Teilschritte in der richtigen Reihenfolge und zur richtigen Zeit in personalisierte Arbeitslisten einsortiert werden. Die Zuordnung von Programmen ermöglicht weiterhin auch eine automatisierte Abarbeitung der Teilschritte. Zuletzt wird dadurch eine nachvollziehbare Überwachung der im Unternehmen ablaufenden Prozesse ermöglicht. Alle vom WFMS benötigten beziehungsweise produzierten Daten wie etwa Prozessbeschreibungen, die Verwaltung des aktuellen Bearbeitungsfortschritts, oder historische Aufzeichnungen der erledigten Teilschritte inklusive den jeweiligen Verantwortlichen, werden unter dem Begriff der Prozesskontrolldaten zusammengefasst.

Innerhalb eines Unternehmens existiert im Regelfall eine zentral verwaltete Infrastruktur, welche zur Bereitstellung eines WFMS und für die Verwaltung der Prozesskontrolldaten genutzt werden kann. Wenn hingegen mehrere Organisationen an einem zwischenbetrieblichen Prozess beteiligt sind, kann im allgemeinen Fall keine zentralisierte Infrastruktur angenommen werden. Eine Alternative bieten hierbei dezentrale Peer-to-Peer-Netzwerke, welche zum Beispiel auch in Blockchain-Protokollen zum Einsatz kommen. Darin verwalten alle Teilnehmer jeweils lokal den gleichen Datenstand, welcher durch geeignete fehlertolerante Algorithmen stets synchronisiert wird. Die Verwendung der dezentral verwalteten Prozesskontrolldaten in jeweils lokal administrierten WFMSs ermöglicht die Ausführung eines zwischenbetrieblichen Prozessmodells inklusive der Möglichkeiten zur Überwachung und kollaborationsweiter Koordination von Teilschritten, ähnlich der traditionellen Auffassung der Prozessausführung innerhalb eines Unternehmens.

In dieser Arbeit werden unter Verwendung der Forschungsmethodik *Design Science Research (DSR)* zwei Artefakte vorgestellt. In einem ersten DSR-Zyklus wird ein Artefakt zur dezentralen Prozessausführung auf Basis einer Blockchain-Infrastruktur vorgestellt. Im Gegensatz zu bestehenden Ansätzen ist die Systemarchitektur des neuen

Artefakts einem WFMS nachempfunden und unterstützt so erstmals grundlegende Funktionalitäten wie die kollaborationsweite Koordination von Arbeitsschritten bei der dezentralen Prozessausführung. Das Artefakt ist allerdings auf die Verwendung von BPMN als Modellierungssprache sowie auf Ethereum als Protokoll zur dezentralen Prozesskontrolldatenverwaltung beschränkt. Darauf aufbauend wird in einem zweiten DSR-Zyklus diese hohe Kohärenz der Domänen Prozessausführung und dezentrale Datenverwaltung bei Blockchain-basierten Ansätzen als Herausforderung identifiziert. Um dieser Kopplung entgegenzuwirken wird eine konzeptionelle Architektur zur Trennung der Prozessausführung und der Kontrolldatenverwaltung als Artefakt vorgestellt. Anstatt dedizierte Artefakte für Prozessbeschreibungssprachen und dezentrale Netzwerkimplementierungen entwickeln zu müssen erlaubt diese Architektur die Integration bereits bestehender Systeme: Über eine Middleware können existierende WFMSs und Algorithmen zur dezentralen Datenverwaltung flexibel integriert werden, wobei auch Alternativen zu Blockchain-Protokollen unterstützt werden. Die Umsetzbarkeit der konzeptionellen Architektur wird schließlich in einer prototypischen Implementierung demonstriert.

ABSTRACT

The research area of business process management comprises the modeling of business processes, the implementation and execution thereof in execution systems as well as the analysis and optimization of processes. In this context, business processes describe recurring operations in enterprises. Through software-supported execution, the process models and included work items are actively interpreted in workflow management systems (WFMS). This allows, for example, the coordination of tasks to be completed across all employees by sorting the work items into personalized work lists in the correct sequence and at the right time. The assignment of programs further enables the automated execution of partial steps. Finally, this allows for traceable monitoring of the processes taking place within the company. All data required or produced by the execution system, such as process descriptions, management of current processing progress, or historical records of completed work items including the respective responsible employees, are summarized under the term process control data.

Within a company, there is usually a centrally managed infrastructure that can be used for the provision of a WFMS and for managing process control data. However, when multiple organizations are involved in an inter-organizational process, a centralized infrastructure cannot generally be assumed. In these settings, decentralized peer-to-peer networks such as blockchain protocols can be used. Thereby, all participants manage the same data locally, which is continuously synchronized through appropriate fault-tolerant algorithms. The use of decentralized process control data in locally administered WFMS enables the execution of an inter-organizational process model, including the ability to monitor and coordinate collaboration-wide work items, similar to the traditional conception of process execution within a company.

In this research work, two artifacts are introduced using the *Design Science Research (DSR)* methodology. In the first DSR cycle, an artifact for decentralized process execution based on a blockchain infrastructure is presented. Unlike existing approaches, the system architecture of the new artifact is modeled after a WFMS, thus supporting for the first time fundamental functionalities such as the collaboration-wide coordination of work steps in decentralized process execution. However, the artifact is limited to the use of BPMN as a modeling language and Ethereum as a protocol for decentralized process control data management. The second DSR cycle identifies the high coherence of the domains of process execution and decentralized data management

in blockchain-based approaches as a challenge. To counter this coupling, a conceptual architecture for separating process execution and control data management is presented as an artifact. Instead of having to develop dedicated artifacts for process description languages and decentralized network implementations, this architecture allows the integration of existing systems: Existing WFMS and algorithms for decentralized data management can be flexibly integrated via a middleware, with support also for alternatives to blockchain protocols. The feasibility of the conceptual architecture is finally demonstrated in a prototype implementation.

INHALTSVERZEICHNIS

ABBILDUNGSVERZEICHNIS

TABELLENVERZEICHNIS

QUELLTEXTVERZEICHNIS

AKRONYMVERZEICHNIS

BFT byzantinische Fehlertoleranz

BPM Business Process Management

BPMN Business Process Management and Notation

DMN Decision Model and Notation

DSR Design Science Research

EVM Ethereum Virtual Machine

ICB Internal Communication Bus

SCI Secure Communication Infrastucture

WFMS Workflow-Management-System

Teil I

PROLEGOMENA

1

EINLEITUNG

"Die Welt besteht aus Daten und Prozessen."

– Stefan Jablonski, im Zeitraum des Betreuungsverhältnisses.

1.1 DATEN UND PROZESSE

Daten und Prozesse

Axiomatisch ist die Welt selbstverständlich weitaus komplexer zu betrachten als sie ausschließlich als Komposition von Daten und Prozessen zu erfassen, und dennoch spiegelt dieses Zitat aus der Zeit der Betreuung dieser Dissertation eine interessante abstrakte Sichtweise zumindest auf die Welt der Informatik wider. Daten dienen dabei als grundlegende beschreibende Bausteine zur statisch strukturellen Darstellung von Sachverhalten, während Prozesse als dynamische Komponente die Daten erzeugen, verändern oder löschen und damit einen temporalen Aspekt in diese Welt integrieren. Beispielsweise können zwei Daten $x_1 = 99$ und $x_2 = 97$ fiktive Bewertungen einer Forschungsarbeit darstellen, während mit einem sehr rudimentären Prozess der Mittelwertbildung das konsolidierte Resultat als neues Datum $x_3 = 98$ ermittelt werden kann. Die Annahme weitaus komplexerer Datenstrukturen und insbesondere Abläufe mit komplexeren Verhaltensmustern lassen die Mächtigkeit einer Welt aus Daten und Prozessen erahnen.

Daten und Prozesse in dieser Arbeit

Der primäre Fokus dieser Arbeit liegt auf Prozessen sowie deren Beschreibung und Durchführung. Dabei werden hier vornehmlich Geschäftsprozesse betrachtet, also wiederkehrende Abläufe in Unternehmen oder Organisationen wie Bewerbungsprozesse, maschinelle Produktionsprozesse oder Lieferkettenprozesse. Alternative Einsatzbereiche sind nicht kategorisch ausgeschlossen, jedoch ist die Anwendbarkeit der in dieser Arbeit vorgestellten Konzepte dort separat zu evaluieren. Obwohl auch die über die Geschäftsprozesse organisierten operativen Daten von Unternehmen, so wie Bewerberdaten, Maschineneinstellungen oder Informationen zu Bestellungen, in dieser Arbeit mitbetrachtet werden, sind die Meta-Daten oder Prozesskontrolldaten von besonderer Bedeutung. Diese umfassen die Da-

ten zur Beschreibung der Prozesse selbst, deren historische Aufzeichnung während der Ausführung, den aktuellen Bearbeitungsfortschritt und damit auch die Organisation der zukünftigen Abarbeitung.

In den folgenden Kapiteln wird das Geschäftsprozessmanagement als Forschungsbereich dieser Arbeit kurz vorgestellt (Kapitel 1.1.1) und dabei auf die besonderen Herausforderungen der zwischenbetrieblichen Prozessausführung eingegangen (Kapitel 1.1.2). Ebenfalls wird speziell die dezentrale Speicherung und Organisation der Prozesskontrolldaten beschrieben (Kapitel 1.1.3), welche insbesondere im zwischenbetrieblichen Kontext eine hohe Relevanz hat. Schließlich werden Blockchain-basierte Systeme vorgestellt (1.1.4), welche als eine Implementierungsmöglichkeit der dezentralen Prozesskontrolldatenverwaltung dienen.

1.1.1 *Geschäftsprozessmanagement*

Einführung Geschäftsprozessmanagement

Der dieser Arbeit zugrundeliegende Forschungsbereich Business Process Management (BPM) oder Geschäftsprozessmanagement thematisiert unter anderem die strukturierte Modellierung von wiederkehrenden Abläufen oder Prozessen, die Implementierung dieser Prozesse in sogenannten Workflow-Management-System (WFMS) sowie die systemgestützte Ausführung mit diesen [39]. Die WFMS-basierte schrittweise Abarbeitung der im Prozess definierten Schritte erfolgt dabei entweder unter Einbeziehung der beschäftigten Mitarbeitenden, welche abzuarbeitende Teilschritte in ihren personalisierten Arbeitslisten finden, oder aber automatisiert durch vordefinierte Programme oder Skripte [78]. Dadurch kann einerseits eine konforme Durchführung in Bezug auf das vordefinierte Modell sichergestellt werden, während andererseits durch Automatisierungsschritte eine effizientere Abarbeitung ermöglicht wird. Zuletzt umfasst das Geschäftsprozessmanagement auch den Teilbereich des *Process Mining*, worin historisch aufgezeichnete Ereignisprotokolle, etwa von WFMSs, aber auch von anderen IT-Systemen, analysiert werden, um unter anderem strukturierte Prozessmodelle automatisiert daraus zu extrahieren oder bereits existierende Prozessmodelle zu optimieren [3]. In Bezug auf den Forschungsbereich des Geschäftsprozessmanagements beschäftigt sich diese Arbeit insbesondere mit der Ausführung von Prozessmodellen zur koordinierten Abarbeitung der darin erfassten Teilschritte.

1.1.2 *Zwischenbetriebliche Prozessausführung*

zwischenebtriebliche Prozessausführung

Die zwischenbetrieblichen Prozesse betrachten im Allgemeinen Interaktionen oder Beteiligungen von verschiedenen Unternehmen oder Organisationen an einem Prozess, also nach obiger Auffassung an einem strukturierten Ablauf zum Erzeugen, Verändern oder Löschen

von Daten, verwaltet durch die Prozesskontrolldaten in einem WFMS. Im Gegensatz zur Prozessausführung innerhalb eines Unternehmens, wo üblicherweise ein WFMS auf einer zentral verwalteten Infrastruktur zur Koordination der Teilschritte bereitgestellt wird, bieten sich im zwischenbetrieblichen Umfeld stattdessen heterogene Umsetzungsmöglichkeiten mit komplexeren Herausforderungen, welche sowohl von einer organisatorischen als auch aus einer technischen Sichtweise betrachtet werden können.

organisatorische Sichtweise

Aus einer organisatorischen Sichtweise heraus kann unterschieden werden, ob ein zwischenbetrieblicher Prozess als konsolidierte Beschreibung aller abzuarbeitenden Teilschritte aufgefasst wird, oder stattdessen als einzuhaltendes Nachrichtenaustauschprotokoll definiert wird, welches lediglich die zu erfolgende Kommunikation unter den beteiligten Organisationen regelt. Dieses Vorgehen sorgt einerseits für ein hohes Maß an Autonomie der Organisationen, indem jeweils interne Arbeitsabläufe unabhängig gestaltet werden können. Andererseits ist eine kollaborationsweite Koordination der Teilschritte und deren globale Nachvollziehbarkeit mit den entsprechenden Verantwortlichkeiten nur eingeschränkt möglich und entspricht nicht dem Konzept der Prozessausführung im engeren Sinne. Des Weiteren sind in der organisatorischen Sichtweise Fragen hinsichtlich des Sicherheitsaspektes zu beantworten, zum Beispiel ob ein gewisses Maß an gegenseitigem Vertrauen besteht oder ob Sicherheitsmaßnahmen gegen Betrugs- oder Manipulationsversuche von den verteilt verwalteten Prozessdaten oder Prozesskontrolldaten zu ergreifen sind.

technische Sichtweise

Die technische Perspektive stellt unabhängig von organisatorischen Fragestellungen ausführende Implementierungen und Infrastrukturen für die Umsetzung zur Verfügung. Danach lassen sich im zwischenbetrieblichen Umfeld ähnlich zur innerbetrieblichen Prozessausführung auch zentral verwaltete Infrastrukturen nutzen. Darüber hinaus ergeben sich durch die logisch disjunkten Netzwerke und Infrastrukturen der beteiligten Unternehmen auch die Möglichkeiten einer verteilten Infrastruktur und einer dezentralen Verwaltung der Daten. Während beide Varianten auf Basis eines Peer-to-Peer-Netzwerks umgesetzt werden können, erfolgt bei einer verteilten Verwaltung die Speicherung der Daten verteilt, während bei der dezentralen Verwaltung die Speicherung repliziert erfolgt, das heißt, jeder Netzwerkknoten speichert dieselben Daten, welche über geeignete Algorithmen stets synchron gehalten werden.

Zusammengefasst werden über die organisatorische Sichtweise eher die Anforderungen der Kollaborationsteilnehmer beschrieben, während die technische Perspektive geeignete Infrastrukturen für die Umsetzung zur Verfügung stellt. Der Fokus in dieser Arbeit liegt dabei auf der technischen Perspektive und damit auf der dezentralen Verwaltung von Prozesskontrolldaten, was in Kapitel 1.1.3 näher erläutert wird.

1.1.3 *Dezentrale Prozessausführung*

dezentrale Prozessausführung

Grundsätzlich sind in einem dezentralen System alle Ressourcen auf mehreren Computern verteilt, wobei geeignete Algorithmen unter anderem für die Kommunikation, Koordination, Fehlertoleranz oder Sicherheit eingesetzt werden [162].

fehlertolerante Kommunikation und Synchronisierung

Demnach lässt sich die dezentrale Prozessausführung im Speziellen darüber definieren, dass die zur Ausführung eines Prozessmodells notwendigen Prozesskontrolldaten, dazu zählen unter anderem die Beschreibung des Prozesses oder die Ausführungshistorie, auf den verteilten Computern der Organisationen repliziert verwaltet werden. Damit alle Benutzer dieses verteilten Systems stets auf einer gleichen Version der Prozesskontrolldaten arbeiten können, müssen dementsprechend alle auftretenden Benutzeranfragen und Ereignisse allen Computern zur Verfügung gestellt werden (Kommunikation [162, Kap. 4]). Weiterhin muss dafür gesorgt werden, dass alle Computer diese Ereignisse in der gleichen Reihenfolge verarbeiten können (Synchronisierung oder Koordination [162, Kap. 5]). Aufgrund der algorithmischen Datensynchronisierung und der fehlenden zentralen Koordinierungsautorität besteht ferner die Gefahr von technisch bedingten oder bewusst manipulierenden fehlerhaften Nachrichten, sodass gegebenenfalls robuste fehlertolerante Algorithmen zu verwenden sind [162, Kap. 8].

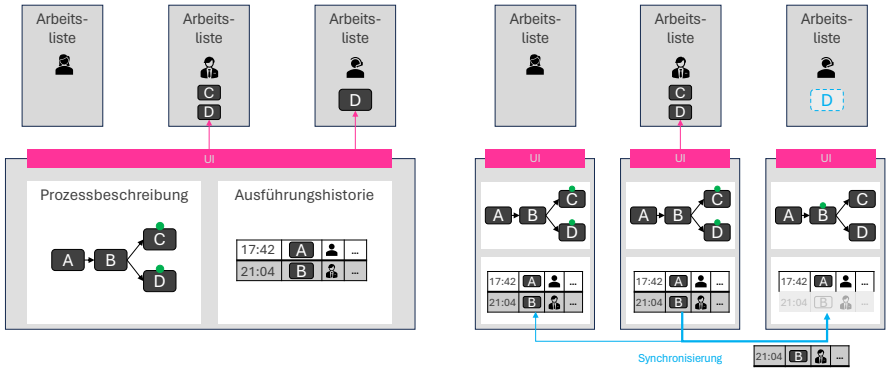

Abbildung 1: Zentrale Verwaltung der Prozesskontrolldaten (links) versus Synchronisierung der dezentral gespeicherten Prozesskontrolldaten (rechts)

Beispiel: Dezentrale Prozessausführung. Abbildung 1 zeigt den Unterschied zwischen zentral verwalteten und dezentral verwalteten Prozesskontrolldaten. Die Prozessbeschreibung, nämlich dass zuerst Schritt A erfolgt, worauf Schritt B folgt, bevor unabhängig voneinander die Schritte C und D den Prozess abschließen, wird links im Bild zentral verwaltet, zusammen mit der bereits erfolgten Ausführungshistorie. Diese zeigt unter anderem die Erledigung von Schritt A um

17:42 Uhr beziehungsweise Schritt B um 21:04 Uhr. Aus diesen Informationen können C und D als nächste Schritte abgeleitet werden, indiziert durch den grünen Punkt. Weiterhin können C und D dann gegebenenfalls unter Einbeziehung zusätzlicher Informationen wie etwa Autorisierungsregeln in den persönlichen Arbeitslisten der Mitarbeitenden registriert werden. Rechts im Bild, bei der dezentralen Verwaltung der Prozesskontrolldaten ist die erforderliche Kommunikation und Synchronisierung bislang nicht abgeschlossen, sodass der Knoten am rechten Bildrand noch eine veraltete Ausführungshistorie annimmt, und dadurch zu diesem Zeitpunkt noch keine aktuelle Arbeitsliste anbieten kann. Dieser Zustand ist jedoch zeitlich begrenzt, sodass eine vollständige Synchronisierung kurzfristig erfolgt.

Mit dieser dezentralen Verwaltung können auch kollaborationsweit Aufgaben koordiniert werden. Im Beispiel sind die Benutzer des mittleren und rechten Knotens beide berechtigt, Verantwortlichkeiten für den Schritt D zu übernehmen. Falls beide Benutzer eine Nachricht zur Beanspruchung der Ausführungsrechte aussenden, sorgt ein Synchronisierungsalgorithmus wiederum für eine global eindeutige Reihenfolge der Nachrichten, sodass der Sender der ersten Nachricht die endgültige Verantwortung zugesprochen bekommt. Zu Vereinfachung abstrahiert in Abbildung 1 die Ausführungshistorie von der Unterscheidung zwischen Beanspruchung und Abschließen eines Teilschritts.

Durch den Einsatz geeigneter Algorithmen kann sich die Kollaboration folglich automatisiert ohne zentrale Verwaltung oder Kontrolle autonom koordinieren.

Im Regelfall ist aus Benutzerperspektive dementsprechend kein Unterschied zwischen der zentral verwalteten und dezentralen Prozessausführung bemerkbar. Allerdings kann im zwischenbetrieblichen Umfeld eine zentral verwaltete Infrastruktur für die Ausführung eines Prozessmodells im allgemeinen Fall nicht angenommen werden, sodass dann auf eine dezentrale Lösung zurückgegriffen werden muss. Dies spiegelt die Trennung zwischen der organisatorischen und technischen Sichtweise wider, da im Grunde auch innerbetriebliche Prozesse auf einer dezentralen Infrastruktur ausgeführt werden können, nämlich wenn unabhängige Abteilungen in einem großen Konzern kollaborieren oder um Vorteile bezüglich Ausfallsicherheit zu nutzen.

Eine Möglichkeit zur Implementierung dezentraler Anwendungen und damit auch der dezentralen Prozessausführung bieten Blockchain-basierte Systeme, welche im Folgenden kurz vorgestellt werden.

1.1.4 *Blockchain-basierte Systeme*

Bitcoin

Anfang der 2010er Jahre erlangte die 2008 veröffentlichte Kryptowährung Bitcoin und die zugrundeliegende Blockchain-Technologie breiteres Interesse, als ein Bitcoin erstmals für mehr als 1 US-Dollar (2011), dann für mehr als 100 US-Dollar (Anfang 2013) und schließlich für mehr als 1000 US-Dollar (Ende 2013) gehandelt wurde.[1] Die Bitcoin-Währung wird von keiner zentralen Organisation, wie beispielsweise staatlichen Institutionen oder Banken, kontrolliert, sondern die Verwaltung erfolgt autonom in einem dezentralen Netzwerk, worin alle finanziellen Transaktionen repliziert verwaltet werden. Die Daten beziehungsweise Transaktionen werden dabei stetig durch die Ausführung innovativer Algorithmen kommuniziert und synchronisiert oder geordnet, wobei jeder Teilnehmer im Netzwerk gleichgestellt ist.

Ethereum

Im Jahr 2013 wurde mit Ethereum ein weiteres Blockchain-Protokoll vorgestellt, welches über monetäre Transaktionen hinaus eine dezentrale Bereitstellung und Ausführung von Computerprogrammen ermöglicht. Das bedeutet, der Programmquelltext wird im Netzwerk als Transaktion bereitgestellt, worauf die Teilnehmer die Ausführung der darin enthaltenen Funktionen über weitere signierte Transaktionen auslösen können. Das Netzwerk einigt sich wiederum stets auf eine globale Ordnung aller Transaktionen, sodass alle Funktionen in den lokalen Anwendungen der teilnehmenden Knoten in derselben Reihenfolge ausgeführt und somit stets dieselben Ergebnisse und Datenstände errechnet werden können.

innovative, disruptive Eigenschaften

Die disruptiven Eigenschaften der weltweiten Netzwerke von Kryptowährungen und Blockchain-Protokollen sind unter anderem durch eine theoretisch unbegrenzte Anzahl von pseudonymisierten Benutzerkonten getrieben. Darüber können unter dem Schutz der Nutzeridentität Geldüberweisungen in Bitcoin durchgeführt werden und theoretisch beliebige Programme und Skripte bereitgestellt werden, dessen Funktionen wiederum alle Netzwerkteilnehmer unter Einhaltung der Pseudonymität ausführen können. Die innovativen Eigenschaften von pseudonymisierten und theoretisch unbegrenzten Benutzerkonten in einem globalen Netzwerk erfordern bestimmte Sicherheitsmaßnahmen, welche durch den Einsatz geschickter Datenstrukturen umgesetzt werden beziehungsweise von Kryptowährungen und deren zugeschriebenen Wert getrieben sind. Dadurch können manipulierende Knoten, welche nicht den Protokollregeln folgen, in monetärer Hinsicht bestraft werden.

Zentralisierung

Aktuelle Blockchain-Protokolle bieten allerdings auch Konfigurationen an, welche die Absicherung anstatt über finanzielle Anreize durch eine stärkere Zentralisierung erreichen. Dabei nehmen bestimmte, vertrauenswürdige und damit oftmals identifizierte Teilnehmer über die Überprüfung und Ordnung von Transaktionen eine Sonder-

1 https://www.btc-echo.de/kurs/bitcoin/, besucht am 28.02.2024

rolle im Netzwerk ein. Blockchain-Protokolle lassen sich durch die unterschiedlichen Konfigurationsoptionen somit in unterschiedlichen Anwendungsszenarien einsetzen.

Blockchain zur dezentralen Prozessausführung

Blockchain-Protokolle basieren auf dezentralen Netzwerken und eignen sich damit für die Implementierung der dezentralen Prozessausführung (Kapitel 1.1.3). Dazu trägt insbesondere die Bereitstellung von Computerprogrammen im Netzwerk bei sowie eine global geordnete Abarbeitungsreihenfolge aller Anweisungen. Die innovativen Konfigurationsmöglichkeiten der Protokolle, welche zum Beispiel die anonyme und theoretisch unbegrenzte Teilnahme erlaubt, stellen Herausforderungen dar. Lösungen und Ansätze werden in dieser Arbeit vorgestellt und diskutiert.

1.2 FRAGESTELLUNG UND ZIEL DER FORSCHUNGSARBEIT

tangierte Forschungsdisziplinen

Die Fragestellungen dieser Dissertation beschäftigen sich mit der Ausführung von Prozessen bei einer dezentralen Verwaltung der Prozesskontrolldaten.

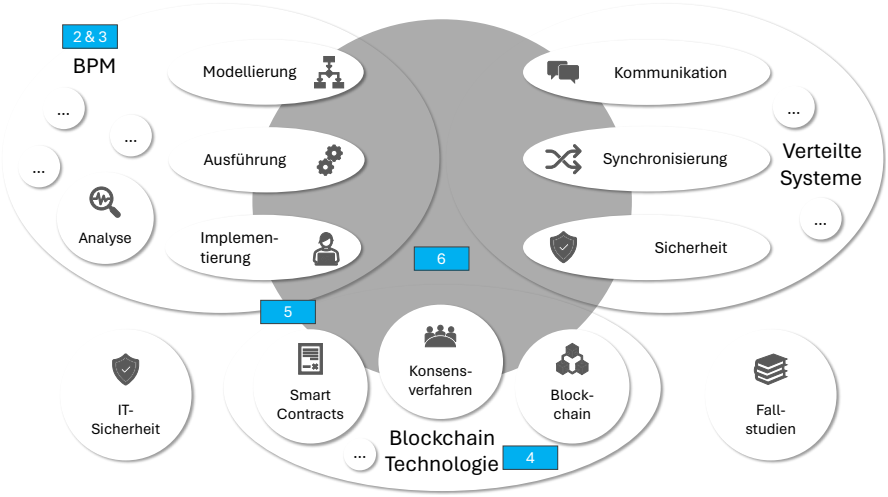

Abbildung 2: Darstellung der Forschungsbereiche und Disziplinen (farblos) und deren Betrachtung in dieser Arbeit (grau) in den entsprechenden Kapiteln (blau)

Wie in Abbildung 2 dargestellt, tangiert diese Arbeit damit primär den BPM-Forschungsbereich, insbesondere die Ausführung von Prozessmodellen, zum Teil aber auch deren Modellierung und Implementierung. Aufgrund des Fokus auf die dezentrale Ausführung müssen konsequenterweise auch die Charakteristika von verteilten Systemen berücksichtigt werden, vordergründig die Kommunikation und Synchronisierung aller Ereignisse. Auswahlalgorithmen können bei der Synchronisierung eine Rolle spielen, wenn ein Knoten für die Festlegung der Reihenfolge von Nachrichten ausgewählt werden muss. Die Blockchain-Technologie ist in Abbildung 2 aufgrund ihrer

innovativen Beschaffenheit sowie in Hinblick auf die Aufteilung dieser Arbeit getrennt von den verteilten Systemen dargestellt. Innerhalb der Blockchain-Technologie sind vor allem die dezentral verwalteten Programme in Form von Smart Contracts von zentraler Bedeutung. Darüber hinaus erfolgt auch eine genaue Betrachtung der Blockchain-Datenstruktur in den Grundlagen und der unterschiedlichen Eigenschaften von Konsensverfahren zur Synchronisierung bei der Prozessausführung. Der BPM-Forschungsbereich wird hauptsächlich in Kapitel 2 und Kapitel 3 betrachtet, während Kapitel 4 Blockchain-Systeme einführt. Kapitel 5 beschreibt die Blockchain-basierte Prozessausführung und Kapitel 6 präsentiert eine flexible Lösung für die Integration von verteilten Systemen und Blockchains für die dezentrale Ausführung von Prozessen.

FORSCHUNGSFRAGEN Die Forschungsfragen, mit denen sich diese Dissertation im Kern beschäftigt, lassen sich wie folgt formulieren.

Interpretation von Prozessmodellen mittels Blockchain-Protokollen

Die erste Forschungsfrage ergibt sich aus den zu Beginn dieser Forschungsarbeit existierenden ersten Ansätze zur Blockchain-basierten Prozessausführung. Innerhalb dieser Prototypen erfolgt die Umsetzung nicht über die im Forschungsbereich übliche dynamische Interpretation von Prozessbeschreibungen zur Laufzeit. Stattdessen werden über den sogenannten kompilierenden Ansatz prozessmodellspezifische Artefakte in Form von Smart Contracts erzeugt, welche den zu erfolgenden Ablauf statisch kodieren. In dieser Arbeit werden die Nachteile dieses Vorgehens aufgearbeitet und darauf basierend mit dem interpretierenden Ansatz eine Lösung vorgeschlagen. Ausgehend von den existierenden Ansätzen [53, 110, 185] liegt der Fokus dabei auf dem Business Process Management and Notation (BPMN)-Standard zur Beschreibung von Prozessen, der breite Verwendung im akademischen und industriellen Umfeld findet [25]. Da die Anwendbarkeit von BPMN dadurch bereits gezeigt ist, stützt sich auch die hier vorgestellte Architektur auf diese Möglichkeit der Beschreibung für Prozesse. Analog ist in den existierenden Ansätzen die Anwendbarkeit der Ethereum-Blockchain [190] als dezentrale Infrastruktur evaluiert, weswegen auch hier erstmal keine neuen Infrastrukturen angebunden werden sollen, sondern speziell der auf Ethereum basierende Ansatz verbessert werden soll. Zusammengefasst beschäftigt sich die erste Forschungsfrage mit der Konzeption eines interpretierenden Ansatzes zur Ethereum-basierten Ausführung von BPMN-Prozessmodellen und lässt sich wie folgt formulieren:

- **RQ1:** Wie kann eine interpretierende Ausführung von Geschäftsprozessen, welche über BPMN-Prozessmodelle definiert sind, auf der Ethereum-Blockchain umgesetzt werden?

Herausforderungen der Blockchain-basierten Prozessausführung

Als Grundlage in Bezug auf die zweite Forschungsfrage dient eine kritische Beurteilung von Blockchain-basierten Ansätzen. Bei den Ansätzen zur Blockchain-basierten Interpretation von Prozessmodellen

erfolgt über die Smart Contracts eine Reimplementierung der Funktionalität von bereits existierenden WFMSs, die üblicherweise im Kontext einer zentral verwalteten Prozessausführung zum Einsatz kommen. Dies hat unter anderem einerseits zur Folge, dass aktuell nur rudimentäre Anforderungen aus der BPM-Domäne umgesetzt werden. Andererseits muss für jedes weitere Blockchain-Protokoll, welches als Alternative zur Ethereum-Blockchain[2] zur dezentralen Prozessausführung eingesetzt werden soll als auch für jede neue Prozessmodellierungssprache ein neues Artefakt entwickelt werden. Als Grund wird eine fehlende klare Schichtentrennung zwischen der Prozessausführungsfunktionalität im BPM-Bereich und den Herausforderungen von verteilten Systemen bei der Blockchain-basierten Prozessausführung identifiziert. Neben diesen Schwachstellen existieren weitere Herausforderungen bei diesen Ansätzen, welche im Folgenden beschrieben sind.

notwendige Flexibilität im zwischenbetrieblichen Umfeld

Innerhalb der organisatorischen Perspektive der zwischenbetrieblichen Prozessausführung sind zum Zeitpunkt des Schreibens dieser Dissertation noch keine allgemeingültigen Leitlinien eruiert, in Bezug auf welche Modellierungssprache beziehungsweise welches Blockchain-Protokoll oder verteiltes System sich am besten für die dezentrale Prozessausführung eignen. Stattdessen lassen die aktuellen Analysen darauf schließen, dass keine allgemeingültigen Aussagen getroffen werden können, und ähnlich zur innerbetrieblichen Prozessausführung verschiedene Modellierungssprachen und verschiedene Infrastrukturen für verschiedene Anwendungsgebiete besser geeignet sind. Dies steht im direkten Widerspruch zur engen Verzahnung zwischen der BPM-Funktionalität und der Umsetzung als verteiltes System auf Basis eines dezentralen Netzwerks bei der Blockchain-basierten Prozessausführung. Dies wird in der zweiten Forschungsfrage aufgegriffen.

Middleware-basierter Ansatz

Im Rahmen der zweiten Forschungsfrage soll ein Artefakt entwickelt werden, welches eine flexibel konfigurierbare dezentrale Prozessausführung erlaubt. Das Artefakt soll einerseits beliebige Prozessmodellierungssprachen unterstützen können und anderseits beliebige Systeme oder Algorithmen für die dezentrale Verwaltung der Prozesskontrolldaten anbinden können, um gegebenenfalls auf die verschiedenen Anforderungen von unterschiedlichen Kollaborationen reagieren zu können. In diesem Zuge soll die Möglichkeit der Verwendung bestehender externer Systeme eine Reimplementierung von Funktionalitäten verhindern. Anwender sollen mit dem Artefakt somit beispielsweise sowohl einen in BPMN modellierten zwischenbetrieblichen Prozess auf Basis der Ethereum-Blockchain ausführen können als auch einen in YAWL [200] notierten weiteren zwischenbetrieblichen Prozess auf Basis traditionellerer fehlertoleranter Synchronisie-

2 genauer gesagt, als Alternative zu Blockchain-Protokollen, welche die Ethereum Virtual Machine (EVM) nicht integrieren

rungsalgorithmen umsetzen können. Zusammengefasst lässt sich die zweite Forschungsfrage wie folgt formulieren:

- **RQ2:** Wie kann ein zu entwickelndes Artefakt eine flexible dezentrale Prozessausführung umsetzen und dabei unabhängig von bestimmten Modellierungssprachen, Blockchain-Protokollen oder zugrundeliegenden dezentralen Infrastrukturen bleiben?

Abgrenzung der Arbeit

ABGRENZUNG DER ARBEIT Diese Arbeit beschäftigt sich mit der Entwicklung von Artefakten zur Beantwortung der Forschungsfragen und Umsetzung der Anforderungen zur Lösung der identifizierten Herausforderungen. Es werden damit keine Fragen zur zwischenbetrieblichen Prozessausführung hinsichtlich organisatorischer Aspekte beantwortet. Insbesondere werden keine konkreten Fallstudien zur Anwendbarkeit der Artefakte in bestimmten Szenarien durchgeführt. Diesbezüglich wird ebenfalls nicht diskutiert, welche Sicherheitsmaßnahmen für bestimmte Anwendungsfälle zu ergreifen sind. Stattdessen liegt der Fokus auf die Integration bereits evaluierter, universell einsetzbarer Konzepte in verwandten Arbeiten, nämlich BPMN oder die Ethereum-Blockchain, und daraus abgeleitete Anforderungen.

Weiterhin bezieht sich der Begriff der Prozessausführung auf die Interpretation eines vordefinierten Prozessmodells. Der Fokus liegt damit auf der Optimierung durch eine Verbesserung der Verwaltung und Koordination von Prozessen. Dabei kommen Funktionalitäten zum Einsatz wie personalisierte Arbeitslisten oder Möglichkeiten zur Überwachung und Nachvollziehbarkeit. Konkrete anwendungsspezifische Verbesserungen wie Laufzeitoptimierungen von dokumentenlastigen Vorgängen durch Digitalisierung oder den Einsatz innovativer Technologien wie Blockchains [50] sind davon abzugrenzen.

Zuletzt hat diese Arbeit nicht zum Ziel, bestehende Modellierungssprachen hinsichtlich der Eignung im zwischenbetrieblichen Umfeld zu evaluieren oder gar neue Konzepte oder Sprachen für deren Definition vorzuschlagen. Stattdessen sollen vorhandene Konzepte und Sprachen bestmöglich verwendet werden.

1.3 METHODIK

DSR als Forschungsmethodik

Die wissenschaftliche Herangehensweise, die der Forschungsarbeit dieser Dissertation zugrunde liegt, basiert auf einem Ansatz im Bereich des Design Science Research (DSR) [70, 138]. *Design Science* zählt zusammen mit der *Behavioural Science* zu den Paradigmen im Forschungsbereich der Informationssysteme (englisch: *Information Systems*) [70]. Während die Behavioural Science-Disziplin das Ziel formuliert, Verhaltensweisen von Menschen oder Organisationen zu erklären, fokussiert sich der Ansatz des Design Science darauf, die Fähigkeiten von Menschen und Organisationen durch neue, innovative Artefakte zu erweitern [70]. Dementsprechend fokussiert sich auch

diese Dissertation auf den Entwurf und die Entwicklung von Artefakten.

verschiedene DSR-Ansätze

Es existieren verschiedene DSR-Ansätze, unter anderem von Hevner et al. [70], Peffers et al.[138] oder Sein et al. [157]. Diese Arbeit verfolgt speziell die Ansätze nach Hevner et al. [70] und Peffers et al. [138]. Der DSR-Ansatz nach Peffers et al. beschreibt dabei sechs Schritte, welche nacheinander ausgeführt werden. Nach diesen Schritten werden auch Kapitel 5 und Kapitel 6 strukturiert, in denen zwei DSR-Zyklen beschrieben werden. Orthogonal dazu beschreiben Hevner et al. Leitlinien, welche in einem DSR-Projekt einzuhalten sind. Insbesondere die Leitlinien für die Evaluation (Schritt 5 bei Peffers et al.) werden für die Bewertung der Artefakte dieser Dissertation angewendet. Im Folgenden werden die beiden DSR-Ansätze genauer vorgestellt. Zusammengefasst beschreiben diese Ansätze eine wissenschaftliche Vorgehensweise zur strukturierten Entwicklung und Evaluation von Informationssystemen oder Software-Artefakten.

1.3.1 *DSR nach Hevner*

Das Modell von Hevner et al. basiert auf den folgenden sieben *Guidelines* (deutsch: Leitlinien):

1. Entwurf als Artefakt (englisch: *Design as an Artifact*)
2. Problemrelevanz (englisch: *Problem Relevance*)
3. Bewertung des Designs (englisch: *Design Evaluation*)
4. wissenschaftlicher Beitrag (englisch: *Research Contributions*)
5. wissenschaftliche Rigorosität/Sorgfalt (englisch: *Research Rigor*)
6. Design als Suchprozess (englisch: *Design as a Search Process*)
7. Wissensvermittlung (englisch: *Communication of Research*)

Die Leitlinien werden in den folgenden Abschnitten genauer vorgestellt. Dabei wird auch darauf eingegangen, inwiefern diese Leitlinien in dieser Arbeit umgesetzt sind.

Leitlinie 1: Entwurf als Artefakt

1. DESIGN AS AN ARTIFACT Nach dieser Leitlinie zentriert sich der DSR-Prozess um ein zu entwickelndes sogenanntes *Artefakt*. Ein Artefakt lässt sich dabei in eine der folgenden Kategorien einordnen [116]: *Construct* oder *Conceptualization* (deutsch: Konzept), *Model* (deutsch: Modell), *method* (deutsch: Methode) oder *Instantiation* (deutsch: Prototyp). Ein Konstrukt ist etwa die Idee des relationalen Modells unter Verwendung von Tabellen zur Strukturierung der Inhalte von Datenbanken. Ein *Entity-Relationship*-Diagramm könnte dieses Konstrukt als Modell aufgreifen, um bestimmte praxisnahe Her-

ausforderungen auf das Konstrukt des relationalen Modells abzubilden. Eine Methode stellt einen Algorithmus oder bestimmte aufeinanderfolgende Schritte als Leitlinien für die Lösung eines bestimmten Problems dar. Ein Prototyp ist letztlich die Realisierung eines konkreten Artefakts in einem bestimmten Anwendungsszenario, wodurch etwa die Umsetzbarkeit einer Methode demonstriert wird.

In dieser Arbeit werden zwei verschiedene DSR-Projekte beziehungsweise zwei DSR-Zyklen durchlaufen. Diese werden in Kapitel 5 und Kapitel 6 beschrieben. Das in Kapitel 5 vorgestellte Artefakt beschreibt dabei einen konkreten Smart Contract, über den die Prozessausführung auf der Ethereum-Blockchain abgewickelt werden kann, und ist damit als konkrete Instantiierung oder Prototyp einzuordnen. Kapitel 6 hingegen stellt im Kern ein abstraktes Konstrukt zur Konzipierung der dezentralen Prozessausführung vor. Für die Evaluation wird die Umsetzbarkeit des Konstrukts zwar zusätzlich über eine prototypische Entwicklung oder Instanziierung demonstriert, das Artefakt jedoch ist das Konstrukt selbst, welches folglich anstatt des Prototyps zu evaluieren ist.

Leitlinie 2: Problemrelevanz

2. PROBLEM RELEVANCE Es ist essenziell, das Problem zu beschreiben, welches durch das Artefakt aus Schritt 1 adressiert wird. Ein Problem kann als Unterschied zwischen dem *Status quo*, also dem aktuellen Status, und einem gewünschten, definierten Zielstatus über eine Anforderungsanalyse definiert werden. Die Problemlösung kann als *Search Process* (deutsch: Suchvorgang) (Leitlinie 6) aufgefasst werden, um die definierten Anforderungen umzusetzen und damit die Unterschiede zum Zielstatus zu minimieren oder zu eliminieren.

Die Relevanz einer dezentralen Prozessausführung wird in dieser Arbeit nicht explizit an einem Fallbeispiel studiert, stattdessen wird die Relevanz durch die bereits aktive Forschung in diesem Bereich und den darin evaluierten Anwendungsfällen im Rahmen einer Literaturstudie begründet. Die Relevanz der in dieser Arbeit entwickelten Artefakte wird auf Basis einer Problemanalyse beschrieben, wonach die aktuellen Ansätze hinsichtlich bestimmter Kriterien zu optimieren sind.

Leitlinie 3: Bewertung

3. DESIGN EVALUATION Die Evaluation bezieht sich auf die Analyse des Artefakts hinsichtlich bestimmter Metriken wie Nutzbarkeit, Qualität oder Effizienz. Nach Leitlinie 5 *Research Rigor* (siehe unten) müssen hierbei etablierte Evaluationstechniken verwendet werden. Diese umfassen beobachtende Methoden, analytische Methoden, experimentelle Methoden, das Testen als Methode sowie beschreibende Methoden. Konkrete Ansätze dieser Methoden sind beispielsweise Fall- oder Feldstudien (beobachtende Methoden), qualitative Komplexitätsanalysen oder quantitative Leistungs- und Laufzeitmessun-

gen (analytische Methoden), kontrollierte oder simulierte Experimente (experimentelle Methoden) oder Softwaretests (Testen).

Die beschreibenden Methoden sind insbesondere für innovative Artefakte geeignet [70]. Da in dieser Arbeit neue Architekturen und Konzepte als Artefakt entwickelt werden, anstatt bereits existierende Artefakte für bestimmte Anwendungsfälle zu evaluieren, ist die deskriptive Evaluation in dieser Arbeit von besonderer Bedeutung. Dabei werden nach Peffers et al. [138] basierend auf einer aufzubauenden *Knowledge Base* (deutsch: Wissensbasis) Probleme und Herausforderungen aktueller Ansätze herausgearbeitet und Ziele für das zu entwickelnde Artefakt definiert. In der Evaluation wird dann basierend auf einer Demonstration die Umsetzbarkeit und Anwendbarkeit der Ansätze gezeigt, und geprüft, inwiefern die ursprünglich definierten Ziele erreicht werden können. Weiterhin erfolgt die Sicherstellung der Qualität der entwickelten Artefakte beziehungsweise der Prototypen über Softwaretests. Explizit ausgeschlossen in dieser Arbeit sind quantitative Analysen und die Evaluation von konkreten Anwendungsszenarien. Letztere werden lediglich zu Demonstrationszwecken unterstützend eingesetzt.

Leitlinie 4: wissenschaftlicher Beitrag

4. RESEARCH CONTRIBUTIONS Ein DSR-Projekt hat einen klaren Forschungsbeitrag in den Bereichen *Design Artifact* (deutsch: Artefakt), *Foundations* (deutsch: Grundlagen) oder *Methodologies* (deutsch: Methodik) aufzuweisen. Oft ist das Artefakt selbst der Beitrag, allerdings können auch Grundlagen, zum Beispiel die Erweiterung der Wissensbasis oder neue Evaluationsmethoden sowie Evaluationsmetriken durch ein DSR-Projekt erweitert werden.

Diese Arbeit konzentriert sich darauf, die Beiträge als Artefakt eines DSR-Projekts zu präsentieren und zu evaluieren. Dass diese Artefakte ebenfalls die grundlegenden Annahmen im Bereich der zwischenbetrieblichen Prozessausführung beeinflussen oder dass durch diese Artefakte neue Methodiken vorgeschlagen werden, soll nicht ausgeschlossen werden. Dies wird hier jedoch nicht explizit betrachtet oder evaluiert.

Leitlinie 5: wissenschaftliche Sorgfalt

5. RESEARCH RIGOR Forschungsarbeiten sind stets unter Einhaltung wissenschaftlicher Rigorosität zu praktizieren. Dies ist durch die Auswahl geeigneter und anerkannter Methoden unter Beibehaltung der Relevanz umzusetzen, während dagegen der Nutzen einer übermäßigen Formularisierung sorgfältig überprüft werden sollte.

In dieser Arbeit wird zuerst mit den Grundlagen und der Beschreibung der Vorgehensweise aktueller Ansätze eine solide Wissensbasis aufgebaut, worauf sich der Entwurf und die Entwicklung der neuen Artefakte stützt. Weiterhin verwendet diese Arbeit unter anderem mit BPMN, dem Camunda WFMS[3], der Ethereum-Blockchain

3 https://camunda.com/, besucht am 10.04.2024

oder BFT-SMaRt [12], einer Implementierung von byzantinische Fehlertoleranz (BFT)-Algorithmen, im Forschungsbereich bereits etablierte Konstrukte oder Systeme. Für den Entwurf der neu entwickelten Artefakte werden anerkannte Prinzipien und Konstrukte der Softwareentwicklungsdisziplin angewandt, darunter die Trennung von Verantwortlichkeiten durch Modularisierung oder UML-Diagramme für die Beschreibung des Entwurfs.

Leitlinie 6: iterativer Suchprozess

6. DESIGN AS A SEARCH PROCESS Ein DSR-Projekt ist von iterativer Natur und verbessert das erzeuge Artefakt schrittweise. Das theoretische Ziel, das bestmögliche Artefakt für eine gegebene Problemstellung zu finden, ist dabei zu ambitioniert: Bei der Suche nach der besten Infrastruktur für die Ausführung zwischenbetrieblicher Prozesse unter Berücksichtigung unter anderem von Kosten, Sicherheit oder Anwendbarkeit würde der Lösungsraum wohl explodieren. Dementsprechend ist im DSR-Prozess eine *zufriedenstellende* Lösung zu finden, welche für eine feste Klasse von Problemen funktioniert.

In dieser Arbeit werden zwei DSR-Projekte vorgestellt, welche auch als einziges DSR-Projekt, bestehend aus zwei Durchläufen oder Zyklen, betrachtet werden können (Kapitel 1.3.2). Dies spiegelt den iterativen Suchprozess wider, da der erste Zyklus auf Grundlage der Wissensbasis aufbaut, während der zweite Zyklus die neu identifizierten Herausforderungen der Evaluation des ersten Zyklus aufgreift. Weiterhin wird das zu erreichende Ziel oder die zu findende Lösung durch die Erarbeitung klar definierter Anforderungen beschrieben, welche jeweils im Evaluationsschritt diskutiert werden.

Leitlinie 7: Wissensvermittlung

7. COMMUNICATION OF RESEARCH Die Präsentation der Artefakte als Ergebnisse eines DSR-Projekts soll sowohl ein technisch versiertes Publikum erreichen als auch an den Managementbereich gerichtet sein. Diese Arbeit fokussiert die technische Beschreibung von Artefakten zur dezentralen Prozessausführung, welche ebenfalls durch technische Hintergründe motiviert sind. Gleichwohl werden die Artefakte auch im Forschungsbereich des Geschäftsprozessmanagements eingegliedert, wobei zum Teil auch betriebswirtschaftliche Hintergründe berücksichtigt werden.

1.3.2 *DSR nach Peffers*

Der DSR-Ansatz nach Peffers schlägt einen einheitlichen Prozess vor, der einerseits nicht in Widerspruch mit bisherigen DSR-Ansätzen steht und andererseits durch ein mentales Modell die Akzeptanz der DSR-Methodik steigern soll [138].

Der Prozess setzt sich aus den folgenden sechs Schritten zusammen:

1. Identifizierung des Problems & Motivation (englisch: *Problem Identification & Motivation*)
2. Zielsetzungen (englisch: *Objectives of a Solution*)
3. Entwurf & Entwicklung (englisch: *Design & Development*)
4. Demonstration (englisch: *Demonstration*)
5. Bewertung (englisch: *Evaluation*)
6. Kommunikation (englisch: *Communication*)

Der DSR-Prozess nach Peffers ist per definitionem von iterativer Natur, das bedeutet, dass die Erkenntnisse aus dem Evaluationsschritt direkt zur Identifikation neuer oder veränderter Problemstellungen führen. Der DSR-Prozess wird daraufhin mit der aktualisierten Problembeschreibung erneut ausgeführt. Dieses Prinzip liegt auch dieser Arbeit zugrunde, was der folgende Absatz kurz darstellt.

DSR in dieser Arbeit

Zu Beginn des ersten DSR-Zyklus wird als Grundlage eine solide Wissensbasis durch ein gründliches Literaturstudium erarbeitet. Der Kern des Literaturstudiums umfasst die innovativen Ansätze für die Ausführung von zwischenbetrieblichen Prozessen auf Basis der Blockchain-Technologie und deren Einordnung in die bisherigen traditionellen Ansätze. Bei der Untersuchung der Blockchain-basierten Prozessausführung werden dabei konzeptionelle Schwächen der Ansätze identifiziert, welche als Problembeschreibung eines ersten DSR-Zyklus dienen (Schritt 1). Darin werden basierend auf den Herausforderungen die Zielsetzungen definiert (Schritt 2), ein Artefakt entworfen und entwickelt (Schritt 3) sowie dessen Funktionsweise demonstriert (Schritt 4). In der retrospektiven Evaluation wird durch einen deskriptiven Ansatz die erfolgreiche Lösung der initialen Problembeschreibung gezeigt (Schritt 5). Allerdings werden in der Evaluation auch prinzipielle Schwachstellen der aktuellen Auffassung der Blockchain-basierten Prozessausführung festgestellt: Da die Ansätze jeweils speziell für eine Modellierungssprache und für eine bestimmte Blockchain[4] eingesetzt werden können, sind diese Ansätze nicht portierbar, was dazu führt, dass jedes weitere Blockchain-Protokoll und jede neue Prozessmodellierungssprache die Entwicklung eines komplett neuen Artefakts erfordert. Ein zweites grundsätzliches Problem ist, dass die Funktionalität von WFMSs von Grund auf neu implementiert werden muss. Auf Basis dieser Herausforderungen wird die Problembeschreibung eines zweiten DSR-Zyklus formuliert und die obigen Schritte erneut durchlaufen.

Die zwei DSR-Zyklen werden in Kapitel 5 und Kapitel 6 beschrieben, wobei die Schritte nach dem DSR-Ansatz nach Peffers die jeweiligen Unterkapitel strukturieren.

4 genauer gesagt für die ausschließliche Ausführung in der EVM

1.4 AUFBAU DER ARBEIT

Diese Arbeit ist in die folgenden vier Teile gegliedert.

1. Prolegomena
2. Dezentrale Prozessausführung auf der Blockchain
3. Ein Rahmenwerk für die flexible Konfiguration externer Systeme zur dezentralen Prozesskontrolldatenverwaltung
4. Synthese

Kapitel 1: Hinführung zum Thema

TEIL I: PROLEGOMENA Der Prolegomena-Teil beinhaltet die Hinführung zum Thema in Kapitel 1, wobei Kapitel 1.1 die Forschungsgebiete kurz vorstellt und Kapitel 1.2 die Fragestellungen dieser Forschungsarbeit präsentiert und Limitationen aufzeigt. Kapitel 1.3 stellt anschließend DSR als Forschungsmethodik vor, bevor der Aufbau der Arbeit hier in Kapitel 1.4 beschrieben wird.

Kapitel 2: BPM-Grundlagen

In Kapitel 2 werden die Grundlagen des Prozessmanagements vorgestellt. Dies umfasst insbesondere Begriffsdefinitionen in Kapitel 2.1, die Modellierung von Prozessen in Kapitel 2.2, die Ausführung von Prozessen in Kapitel 2.3, die wichtigsten Komponenten von WFMSs in Kapitel 2.4 und die verschiedenen Perspektiven von Prozessmodellen, welche in Kapitel 2.5 aufgearbeitet werden.

Kapitel 3: zwischenbetriebliche Prozessausführung

Kapitel 3 überträgt die eingeführten Grundprinzipien auf die zwischenbetriebliche Prozessausführung. Dafür erfolgt zuerst eine initiale Begriffsbestimmung in Kapitel 3.1. Darauf aufbauend werden zwei verschiedene Umsetzungsparadigmen vorgestellt. Während Kapitel 3.2 dabei die nachrichtenbasierte Synchronisation lokaler Prozesse beschreibt, führt Kapitel 3.3 die prozessbasierte Ausführung zwischenbetrieblicher Prozesse ein. Kapitel 3.4 diskutiert die vorgestellten Ansätze kurz. Zuletzt stellt Kapitel 3.5 die sogenannten Formen von Workflow-Interoperabilität vor, welche Umsetzungsmöglichkeiten der zwischenbetrieblichen Prozessausführung aus verwandten Arbeiten beschreiben. Mit Decentralized Control wird dabei eine weitere Form neu eingeführt.

TEIL II: DEZENTRALE PROZESSAUSFÜHRUNG AUF DER BLOCKCHAIN

Kapitel 4: Blockchain-Grundlagen

Der zweite Teil startet mit den Grundlagen Blockchain-basierte Systeme in Kapitel 4. Darin erfolgt zuerst eine abstrakte Einführung in Blockchains in Kapitel 4.1. Daraufhin werden die wichtigsten mathematischen und formalen zugrundelegenden Konzepte von Blockchain-Systemen in Kapitel 4.2 erklärt. Mit diesen Grundlagen wird in Kapitel 4.3 die Funktionsweise einer Blockchain detailliert beschrieben, bevor in Kapitel 4.4 eine bewertende Zusammenfassung erfolgt.

Kapitel 5: Blockchain-basierte dezentrale Prozessausführung

Kapitel 5 stellt den ersten Durchlauf des DSR-Projekts vor. Das Kapitel wird basierend auf den oben beschriebenen DSR-Schritten nach

Peffers et al. [138] strukturiert. Darin wird zuerst die Kompilierung eines Prozessmodells in einen Prozessmodell-spezifischen Smart Contract als konzeptionelle Strategie zur zwischenbetrieblichen Prozessausführung aktueller Ansätze in Kapitel 5.1 beschrieben. Darauf basierend werden mittels eines Vergleichs zu kompilierenden und interpretierenden Sprachen in der Softwareentwicklung Schwächen und Herausforderungen des kompilierenden Ansatzes bei der Prozessausführung in Kapitel 5.2 identifiziert, insbesondere bei der Ausführung auf Blockchain-basierten Systemen. Folglich definiert Kapitel 5.3 unter anderem eine WFMS-ähnliche, interpretierende Ausführung von Prozessmodellen auf der Ethereum-Blockchain als Anforderung an ein Artefakt. In Kapitel 5.4 wird das entsprechende Artefakt entwickelt und dessen Funktionsweise in Kapitel 5.5 demonstriert. Abschließend evaluiert Kapitel 5.6, ob die definierten Anforderungen durch das entwickelte Artefakt erfüllt werden, und identifiziert dabei noch offene Herausforderungen.

TEIL III: EIN RAHMENWERK FÜR DIE FLEXIBLE KONFIGURATION EXTERNER SYSTEME ZUR DEZENTRALEN PROZESSKONTROLLDATENVERWALTUNG Kapitel 6 beschreibt den zweiten Durchlauf des DSR-Projekts, worin die offenen Herausforderungen des vorangegangenen Zyklus bei der Problembeschreibung aufgegriffen werden. Die Kapitelstrukturierung erfolgt ebenfalls nach Peffers et al. [138] während für die Ausgestaltung des Evaluationsschritts die entsprechende Leitlinie von Hevner et al. [70] herangezogen wird. Speziell wird die enge Verzahnung zwischen dem Ethereum-Protokoll als Blockchain und der Prozessausführung negativ hervorgehoben. Stattdessen soll in Kapitel 6 ein Framework entwickelt werden, welches eine klare Schichtentrennung der BPM-Domäne und den Herausforderungen von verteilten Systemen im Allgemeinen forciert. Folglich sollen neben der Schichtentrennung auch Blockchain-alternative Systeme und Algorithmen für die Prozessausführung durch die Integration in eine konsolidierende Plattform eröffnet werden. Das Framework wird im Rahmen eines DSR-Zyklus entwickelt.

Kapitel 6: dpex-Framework

Zu Beginn verschafft Kapitel 6.1 einen groben Überblick über das Framework. Kapitel 6.2 liefert die DSR-Problembeschreibung, bevor Kapitel 6.3 die Anforderungen an das Framework auflistet, kategorisiert nach der BPM-Domäne, verteilten Systemen, und allgemeinen Entwurfsentscheidungen. Kapitel 6.4 stellt das Framework als Artefakt vor, inklusive des strukturellen Aufbaus, der Funktionsweise und der Verwendung. Kapitel 6.5 stellt den Demonstrationsschritt des DSR-Zyklus dar und beschreibt dabei eine Möglichkeit der implementierungstechnischen Umsetzung des abstrakten Frameworks. Insbesondere wird gezeigt, wie ein spezifisches WFMS und ein bestimmtes Blockchain-Protokoll im Framework zur dezentralen Ausführung von Prozessen integriert werden können. Dabei wird auch

demonstriert, wie das WFMS erweitert werden kann, um etwa Autorisierungsregeln für Prozessschritte basierend auf einer strukturierten Beschreibung der Prozessteilnehmer umzusetzen. Zuletzt erfolgt in Kapitel 6.6 wieder die Evaluation des vorgestellten Artefakts im Rahmen des DSR-Zyklus.

Kapitel 7: Evaluation

TEIL IV: SYNTHESE Nachdem die beiden vorgestellten Artefakte bereits innerhalb des jeweiligen DSR-Zyklus evaluiert sind, fasst Kapitel 7 im letzten Teil die Bewertungen zusammen und stellt den Bezug zu den ursprünglichen Forschungsfragen her. Zuletzt erfolgt ebenfalls eine zusammenfassende Gesamtbewertung dieser Arbeit.

Kapitel 8: Verwandte Arbeiten

Kapitel 8 stellt eine Literaturanalyse von verwandten Arbeiten im Forschungsbereich der Blockchain-basierten Prozessausführung vor. Die Analyse erfolgt unter Verwendung der Schneeball-Methode für systematische Literatur-Übersichtsarbeiten. Dabei werden basierend auf einer Startmenge an Publikationen über eine Vorwärts- und Rückwärtssuche weitere relevante Veröffentlichungen selektiert. Ein Vergleich mit bereits existierenden Übersichtsarbeiten lässt auf eine gute Abdeckung schließen. Kapitel 8.1 beschreibt die verfolgte Methodik im Detail, bevor Kapitel 8.2 die relevantesten Arbeiten im Kern des Forschungsgebiets präsentiert. In Kapitel 8.3 erfolgt eine Abgrenzung der in dieser Arbeit vorgestellten Forschungsbeiträgen von diesen relevantesten Veröffentlichungen, welche im Zuge dessen hinsichtlich konzeptioneller Ansätze, verwendete Prozessmodellierungssprachen oder Kommunikationsinfrastrukturen kategorisiert werden. Im Folgenden werden noch die weiter entfernten Ansätze der Literaturrecherche in Kapitel 8.4 beschrieben und ebenfalls unter anderem in die Kategorien Modellierung, Analyse, Überwachung oder Entscheidungsfindung eingeordnet. Kapitel 8.5 fasst die anderen Literatur-Übersichtsarbeiten zusammen. Kapitel 8.6 beschreibt schließlich die Arbeiten mit entfernten Bezug zum Forschungsbereich dieser Arbeit, bevor Kapitel 8.7 mit den sonstigen selektierten Arbeit abschließt.

Kapitel 9: Bewertende Schlussbemerkungen

Kapitel 9 finalisiert diese Arbeit, indem in Kapitel 9.1 die Arbeit inhaltlich zusammengefasst wird und in Kapitel 9.2 der Beitrag der Forschungsarbeit nochmals explizit dargestellt und von anderen Forschungsbereichen abgegrenzt wird. Das letzte Kapitel 9.3 präsentiert schließlich mögliche Anknüpfpunkte für künftige Forschungsarbeiten.

2

GESCHÄFTSPROZESSMANAGEMENT

In diesem Kapitel werden die essenziellen Grundlagen des Geschäftsprozessmanagements (BPM) vorgestellt. Insbesondere wird erklärt, wie sich reale, sich wiederholende Vorgänge oder Prozesse in Unternehmen mithilfe von Modellen abbilden lassen und wie diese Modelle unterstützend in IT-Systemen eingesetzt werden können.

Für die Modellierung der verwendeten Beispielprozesse wird im Folgenden der BPMN-Standard (*Business Process Model and Notation*) verwendet. Die darin definierten Sprachen und Notationen werden vielfach in Wissenschaft und Industrie eingesetzt und avancierten zum de-facto Standard [25]. In dieser Arbeit wird keine explizite Einführung in BPMN gegeben und stattdessen auf die offizielle Spezifikation des Standards verwiesen [135], worin die syntaktischen und semantischen Regeln im Detail erläutert werden. Die hier verwendeten Modelle und Modellierungselemente werden dennoch auch größtenteils im Fließtext beschrieben.

Die Modellierung und Ausführung von Prozessen ist in dieser Arbeit größtenteils mit BPMN und dem System *Camunda* umgesetzt. Dementsprechend erfolgt die Einführung der Konzepte zuerst allgemein auf einer abstrakten Ebene, während die Erläuterungen zur praktischen Anwendbarkeit stets einen engen Bezug zu BPMN und Camunda aufweisen. So wird beispielsweise die Umsetzung der Prozessperspektiven in Hinblick auf die weiteren Ausführungen hauptsächlich vor dem Hintergrund dieser Artefakte demonstriert.

Im Folgenden werden in Kapitel 2.1 die wichtigsten Begriffe aus dem BPM-Bereich für diese Arbeit eingeführt. Kapitel 2.2 beschreibt kurz die Modellierung, zuerst allgemein, dann mit Fokus auf die Modellierung von Geschäftsprozessen mit BPMN im Kontext dieser Arbeit. Die Ausführung, also die aktive Interpretation der erzeugten Modelle, wird in Kapitel 2.3 beschrieben, wobei WFMSs als geeignete Systeme zur Prozessausführung in Kapitel 2.4 vorgestellt werden. Zuletzt beschreibt Kapitel 2.5 fortgeschrittene Konzepte im Bereich der Prozessmodellierung, indem verschiedene Perspektiven von Prozessmodellen aufgezeigt werden, welche speziell für eine wertschöpfende Ausführung eine hohe Relevanz haben.

2.1 PROZESS, PROZESSMODELL UND PROZESSLEBENSZYKLUS

Geschäftsprozess

Einige Tätigkeiten in einem Unternehmen zeichnen sich durch ein bestimmtes Charakteristikum aus, nämlich dass sich diese Tätigkeiten auf dieselbe oder ähnliche Weise über die Zeit hinweg wiederholt im Unternehmen beobachten lassen. Diese wiederkehrenden Abläufe definieren sich unter anderem über die Teilschritte, welche in ihrer Gesamtheit ein bestimmtes Ziel verfolgen. Wiederkehrende, strukturierte und zielorientierte Abläufe werden auch als Geschäftsprozess bezeichnet [39, Kap. 1.1].

Prozessmodell

Geschäftsprozesse können in Modellen erfasst werden, wobei ein Modell stets ein Abbild eines realen Artefakts ist (Abbildungsmerkmal [161, S. 131]). Ein Geschäftsprozessmodell, kurz Prozessmodell, ist demnach das Abbild eines real existierenden Vorgangs. Modelle im Allgemeinen und insbesondere Modelle von Geschäftsprozessen dienen einem bestimmten Zweck und abstrahieren dabei beliebig aber stets zweckdienlich von Details des realen Artefakts oder Vorgangs (Verkürzungsmerkmal [161, S. 131]).

Prozessinstanz

Während das Prozessmodell als abstraktes Abbild versucht, die relevanten Details der Gesamtheit aller einzelnen Vorgänge zu erfassen, wird ein konkreter Vorgang eines Prozesses auch als Prozessinstanz oder Instanz des Prozessmodells bezeichnet [187, S. 90–94]. Verschiedene Instanzen von einem Prozessmodell können sich beispielsweise in den darin verarbeiteten Daten oder bezüglich den beteiligten Personen unterscheiden. Auch die einzelnen Teilschritte, welche konkret ausgeführt werden, können sich bei verschiedenen Instanzen unterscheiden. Das Prozessmodell legt hierbei fest, nach welchen Regeln die Teilschritte ausgewählt und ausgeführt werden können.

Zweck von Prozessmodellen

Modelle werden nach der allgemeinen Modelltheorie stets speziell für einen bestimmten Zweck erstellt [161, S. 132–133]. Modelle von Geschäftsprozessen werden beispielsweise für die Dokumentation in Handbüchern verwendet, als Kommunikationsmedium in Planungsphasen von Softwareprojekten oder sie werden zur Ausführung in einem Softwaresystem eingesetzt. Für die Dokumentation ist eine menschenlesbare Form eines Modells zu bevorzugen, sodass sich zum Beispiel neue Arbeitnehmer an der Dokumentation orientieren können, um die Abläufe im Unternehmen kennenzulernen. Wenn Prozessmodelle für den Einsatz zur softwaregestützten Ausführung in einem sogenannten WFMS oder Prozessausführungssystem vorbereitet werden, ist eine eher strukturierte, formale und maschinenlesbare Form des Modells zu bevorzugen.

Prozesslebenszyklus

Der Forschungsbereich des Geschäftsprozessmanagements umfasst neben der Prozessmodellierung noch weitere Teilbereiche, unter anderem die Ausführung oder die Analyse von Prozessen. Diese Teilbereiche zeigen die zeitlich aufeinanderfolgenden Phasen, welche ein Geschäftsprozess in einem Unternehmen durchläuft, und werden im

sogenannten Prozesslebenszyklus (englisch: *Process Life Cycle*) zusammengefasst. Es existieren verschiedene Varianten von Prozesslebenszyklen, welche sich in der Benennung und Aufteilung von Phasen unterscheiden können oder je nach Kontext weitere Phasen mitbetrachten. Als Grundlage für diese Arbeit wird der Prozesslebenszyklus vereinfacht aber ausreichend über vier Phasen definiert, welche sich so grundsätzlich auch in anderen Modellen wiederfinden lassen (van der Aalst et al. [3, S. 31], Dumas et al. [39, S. 23] oder Weske et al. [187, S. 12]). Abbildung 3 zeigt die vier Phasen, welche im Folgenden näher erläutert werden.

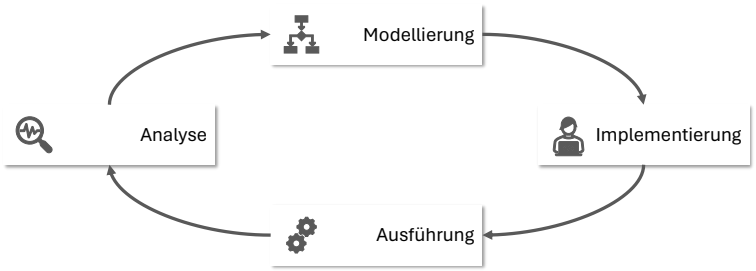

Abbildung 3: Die vier Phasen eines Prozesslebenszyklus

Phase 1: Modellierung

MODELLIERUNG Die Modellierungsphase dient zur Erstellung eines Prozessmodells, einem strukturierten Artefakt, welches einen bestimmten Ablauf im Unternehmen repräsentiert. Dieses Modell ist nach aktuellem Kenntnisstand die optimale Variante des Geschäftsprozesses und wird in der Implementierungsphase weiterverarbeitet. Bei der Modellierung können sowohl die Erkenntnisse aus vorangegangenen Prozessausführungen mit einfließen als auch anderweitig definierte Anforderungen berücksichtigt werden. Die Modellierung wird im Detail in Kapitel 2.2 beschrieben.

Phase 2: Implementierung

IMPLEMENTIERUNG Das Prozessmodell wird anschließend für die spätere Ausführung vorbereitet. Dies umfasst unter anderem die Anbindung an bestehende Systeme oder Datenbanken, um die Erstellung oder Aktualisierung von Daten und Dokumenten zu ermöglichen [39, Kap. 10.5.2]. Auch die Definition von REST-API-Endpunkten, mit welchen während der Prozessausführung kommuniziert werden soll, werden in der Implementierungsphase definiert [39, Kap. 10.5.3], ebenso wie zusätzlicher Quelltext für automatisierte Berechnungen während der Prozessausführung [39, Kap. 10.5.5].

Phase 3: Ausführung

AUSFÜHRUNG Während der Ausführung sorgt ein Softwaresystem für die korrekte Abarbeitung der im Prozessmodell definierten Teilschritte und die Bereitstellung für personalisierte Arbeitslisten für beteiligte Personen [39, Kap. 9.1.3]. Das Softwaresystem

produziert außerdem ein sogenanntes Ereignisprotokoll (englisch: *Event Log*), welches der Reihe nach alle aufgetretenen Aktionen enthält. Dies umfasst die Reihenfolge aller ausgeführten Teilschritte, gegebenenfalls zusammen mit einem Zeitstempel, den beteiligten Personen oder Datenwerten. Die Ausführung wird im Detail in Kapitel 2.3 beschrieben.

Phase 4: Analyse

ANALYSE In der Analysephase wird das während der Ausführung erstellte Ereignisprotokoll dazu verwendet, mögliche Schwachstellen im aktuellen Prozessmodell zu erkennen. Dabei können zum Beispiel zeitliche Flaschenhälse identifiziert werden, die durch eine mögliche Parallelisierung von Teilschritten aufgelöst werden können. Die Ergebnisse der Analysephase werden dann dazu verwendet, das aktuell implementierte Prozessmodell zu verbessern, das heißt, ein überarbeitetes Prozessmodell zu erstellen, womit sich der Kreis in der Modellierungsphase schließt. Die Analyse von Prozessmodellen wird in dieser Arbeit nicht weiter betrachtet. Eine umfassende Einführung in das Thema bietet van der Aalst et al. [3].

Die Modellierung und vor allem die softwaregestützte Ausführung von Prozessen sind in dieser Arbeit von zentraler Bedeutung und werden in den folgenden Kapiteln näher erläutert.

2.2 MODELLIERUNG DER PROZESSE UND BPMN

Das Ziel der Modellierungsphase ist die Erstellung eines Prozessmodells, das bedeutet, einen real existierenden Ablauf in einem Unternehmen in eine abstrakte, modellhafte Form zu bringen. Wie im Folgenden näher beschrieben, kann dies unter Verwendung einer freien Notation erfolgen oder mithilfe von imperativen oder deklarativen Modellierungssprachen.

freie Notation

Die freie Notation überzeugt durch die Flexibilität und Freiheit, welche es dem Prozessmodellierer erlaubt, sehr spezifische und komplexe Inhalte darzustellen und durch beliebige Notationselemente zu verbildlichen. Frei notierte Prozessmodelle haben jedoch eine sehr geringe Reichweite, da die benutzerdefinierten Symbole nicht zwangsläufig von anderen Personengruppen verstanden werden. Weiterhin eignen sich frei notierte Prozessmodelle nicht für die unten beschriebene softwaregestützte Ausführung und dienen daher eher Dokumentationszwecken.

Modell und Sprache

MODELLIERUNGSSPRACHEN Generell ist die Verwendung einer Modellierungssprache zu bevorzugen und darüber hinaus obligatorisch für die spätere Verwendung zur Prozessausführung. Modelle im Allgemeinen zeichnen sich dadurch aus, Sachverhalte durch Abstraktion verständlich darzustellen. Klar definierte Modellierungssprachen

ermöglichen dabei die Erstellung von Modellen mit einem global gemeinsamen Verständnis aller Benutzer. Jede Person, welche eine bestimmte Sprache versteht, hat somit die Möglichkeiten, alle Prozessmodelle zu verstehen, welche in dieser Sprache modelliert sind.

Syntax, Semantik und Metamodell

Dazu spezifiziert eine Modellierungssprache unter anderem eine Syntax, ein Metamodell und die Semantik [179]. Die Syntax definiert Notationselemente als textuelle oder bildhafte Bausteine, etwa einen Kreis, ein abgerundetes Rechteck oder einen Pfeil. Diese Elemente werden dann zur Erstellung eines Modells verwendet. Die Semantik einer Modellierungssprache beschreibt die Bedeutung der Elemente und damit, wie sich das Modell zur Zeit der Ausführung verhält. Zum Beispiel kann der Kreis den Start eines Prozesses verdeutlichen oder das Rechteck wird für die Definition eines Teilschritts oder einer Aktivität verwendet. Wenn ein Pfeil zwei Rechtecke beziehungsweise Aktivitäten verbindet, muss die erste Aktivität beendet sein, bevor die zweite Aktivität startet. Das Metamodell definiert schließlich die Struktur gültiger Modelle und schränkt die Möglichkeiten ein, auf welche Weise die Modellierungselemente miteinander verknüpft werden dürfen. So kann das Metamodell eine gerichtete Abhängigkeit ausgehend von einem Rechteck (Aktivität) hin zu einem Kreis (Start des Prozesses) verbieten, da der Start grundsätzlich den Beginn des Prozesses beschreiben muss.

grafische vs. textuelle Syntax

Der in dieser Arbeit verwendete BPMN-Standard spezifiziert sowohl eine grafische als auch eine textuelle Syntax. Dadurch können BPMN-Modelle einerseits in einer grafischen, für Menschen gut lesbaren Form repräsentiert werden und andererseits in einer textuellen Form, welche die Interpretation der Modelle für Maschinen und Applikationen bei der Prozessausführung vereinfachen (Kapitel 2.3).

Durch die Metamodelle ermöglichen Modellierungssprachen weiterhin eine softwaregestützte Hilfestellung in Form von Editoren. Diese können Modelle bezüglich des Metamodells auf Konsistenz oder Fehler prüfen oder mit Vorschlägen die Modellierung unterstützen. Die Definition der Semantik erlaubt weiterhin die Implementierung von Ausführungssystemen und damit auch die softwaregestützte Ausführung von Prozessmodellen.

PARADIGMEN VON PROZESSMODELLIERUNGSSPRACHEN Prozessmodellierungssprachen können in imperative oder deklarative Sprachen kategorisiert werden [165]. Bei imperativen Sprachen wie BPMN sind alle möglichen Ausführungspfade explizit im Modell definiert. Deklarative Sprachen hingegen schaffen durch die Definition von Bedingungen einen Rahmen, in welchem sich die Prozessausführung bewegen darf. Innerhalb dieses Rahmens ist die Ausführung flexibel, jedoch dürfen die Bedingungen und Einschränkungen des gegebenen Rahmens nicht verletzt werden. Für Routineprozesse, welche stets ähnlich ablaufen und wenig Entscheidungspunkte haben, sind eher

imperativ vs. deklarativ

imperative Sprachen geeignet. Prozesse mit flexibler Ausführung wie die Behandlung von Patienten in Krankenhäusern können eher mithilfe einer deklarativen Sprache ausgedrückt werden [165]. Im Allgemeinen ist es möglich, Prozesse mit Sprachen beider Kategorien zu modellieren. Imperative Modelle von flexiblen Prozessen tendieren jedoch dazu, zu sehr komplexen Modellen mit vielen verschiedenen Kontrollflüssen, sogenannten Spaghetti-Modellen [3, Kap. 14], zu entarten. Die deklarative Beschreibung von einfachen Kontrollflüssen kann hingegen die Definition mehrerer Regeln erfordern.

Es existieren somit Gründe für die Existenz, Entwicklung und den Einsatz sowohl von imperativen Modellierungssprachen als auch deklarativen Modellierungssprachen.

BPMN BPMN steht für *Business Process Model and Notation* und ist ein weitverbreiteter Standard in Industrie und Forschung. Der BPMN-Standard ist eine Sammlung von mehreren Diagrammen oder Sprachen, die unterschiedliche Anwendungsfälle und Zielarchitekturen fokussieren [135, S. 21].

BPMN Prozessdiagramme

Die *Private (Internal) Business Processes* (deutsch: privaten unternehmensinternen Geschäftsprozesse) oder auch die *BPMN Process Diagrams* (deutsch: Prozessdiagramme) werden üblicherweise für die Dokumentation unternehmensinterner Prozesse und deren orchestrierte Ausführung eingesetzt. Das bedeutet, dass die Prozessausführung mithilfe eines zentral verwalteten Softwaresystems erfolgt.[1] Das BPMN-Modell in Abbildung 4 zeigt ein Beispiel eines Prozessdiagramms. Die Bedeutung der Notationselemente ist im Grunde im bisherigen Abschnitt bereits erläutert: Der Kreis auf der linken Seite stellt ein *Start Event* (deutsch: Startereignis) dar und denotiert damit den Beginn des Prozesses. Die Pfeile dazwischen repräsentieren den *Sequence Flow* (deutsch: Kontrollfluss). Sie geben die Flussrichtung des Prozesses an und damit die Abarbeitungsreihenfolge der *Tasks* (deutsch: Aktivitäten), welche in den abgerundeten Rechtecken visualisiert sind. Der Prozess wird mit dem *End Event* (deutsch: Endereignis) beendet.

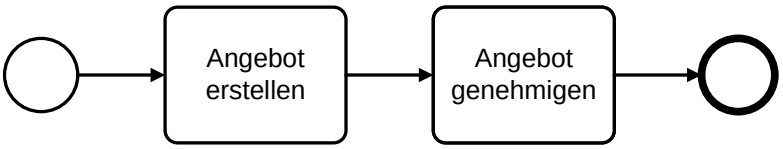

Abbildung 4: Beispielprozess in BPMN

Abbildung 5 zeigt das Prozessmodell aus Abbildung 4 in der textuellen XML-Struktur. Darin lassen sich die oben eingeführten BPMN-

1 Der Standard unterscheidet dabei zwischen ausführbaren (englisch: *executable*) und nicht ausführbaren (englisch: *non-executable*) Prozessen. Bei letzteren wird entweder aufgrund eines reinen Dokumentationszwecks von wichtigen Ausführungsdetails abstrahiert oder diese sind bisher nicht bekannt und werden zu einem späteren Zeitpunkt ergänzt.

Elemente identifizieren, die über sogenannte `sourceRefs` (deutsch: Quellreferenz) und `targetRefs` (deutsch: Zielreferenz) miteinander verbunden sind.

```
<bpmn:process id="angebot" isExecutable="true">
    <bpmn:startEvent id="se1">
        <bpmn:outgoing>Flow_1</bpmn:outgoing>
    </bpmn:startEvent>

    <bpmn:task id="a1" name="Angebot erstellen">
        <bpmn:incoming>Flow_1</bpmn:incoming>
        <bpmn:outgoing>Flow_0</bpmn:outgoing>
    </bpmn:task>

    <bpmn:sequenceFlow id="Flow_1"
        sourceRef="se1" targetRef="a1" />

    <bpmn:task id="a2" name="Angebot genehmigen">
        <bpmn:incoming>Flow_0</bpmn:incoming>
        <bpmn:outgoing>Flow_1</bpmn:outgoing>
    </bpmn:task>

    <bpmn:sequenceFlow id="Flow_0"
        sourceRef="a1" targetRef="a2" />

    <bpmn:endEvent id="ee">
        <bpmn:incoming>Flow_1</bpmn:incoming>
    </bpmn:endEvent>

    <bpmn:sequenceFlow id="Flow_1"
        sourceRef="a2" targetRef="ee" />
</bpmn:process>
```

Abbildung 5: XML-Repräsentation des Beispielprozesses in BPMN

weitere BPMN Diagramme

Der BPMN-Standard stellt neben den Prozessdiagrammen noch weitere Diagrammarten zur Verfügung. Dazu zählen das *Collaboration Diagram* (deutsch: Kollaborationsdiagramm) und die *BPMN Choreography* (deutsch: Choreografie), welche eher im zwischenbetrieblichen Umfeld Anwendung finden und daher in dieser Arbeit besonders von Interesse sind. Die Diagrammarten werden in Kapitel 3 eingeführt.

2.3 PROZESSAUSFÜHRUNG

Kapitel 2.2 beschreibt die Zweckdienlichkeit von Prozessmodellen. Prozesse können demnach einen Dokumentationszweck erfüllen oder in einem Softwaresystem zur Prozessausführung eingesetzt werden. Dabei sorgt die Syntax einer Modellierungssprache dafür, dass die Modelle universell gelesen werden können, und durch die Semantik der einzelnen Notationselemente werden die Prozessmodelle weiterhin auch universell verstanden oder interpretiert. Dieses Kapitel geht

nun darauf ein, wie die Semantik einer Prozessmodellierungssprache definiert werden kann. Dadurch wird den Notationselementen eine Bedeutung zugeschrieben, was die Basis für die Implementierung eines Prozessausführungssystems ist.

MÖGLICHKEITEN DER SEMANTIKDEFINITION Es existieren mehrere Möglichkeiten zur Definition der Semantik von Modellierungssprachen. Zum einen kann die Semantik formal auf einer mathematischen Grundlage definiert werden, etwa bei der deklarativen Prozessmodellierungssprache *DECLARE* [139]. DECLARE stellt *Templates* (deutsch: Vorlagen) für Regeln zur Verfügung, mit denen ein Prozess modelliert werden kann.[2] Als Beispiel verknüpft die sogenannte Vorlage *Response* zwei Aktivitäten A und B und drückt damit aus, dass bei der Ausführung von Aktivität A bis zum Ende der Prozessinstanz mindestens einmal B ausgeführt werden muss. Die Vorlagen beziehungsweise die Regeln fundieren in der mathematischen Teildisziplin der linearen temporalen Logik, worin die oben textuell beschriebene *Response*-Vorlage durch die Formel $\Box(A \Rightarrow \diamond(B))$[3] ausgedrückt wird. Eine weitere Möglichkeit bietet die Überführung in ein anderes Modell, welches bereits eine Semantik definiert. Petri-Netze [140], welche unter anderem für die Verhaltensmodellierung verteilter Systeme verwendet werden, besitzen eine Semantik, indem der Übergang von einem Zustand in den nächsten Zustand eindeutig definiert ist. Es ist gezeigt, dass sich BPMN-Prozessdiagramme in ein Petri-Netz transformieren lassen [31, 37]. Über die Simulation des Petri-Netzes kann so der Prozessstatus festgestellt werden und daraus abgeleitet werden, welche Aktivitäten als Nächstes auszuführen sind.

Formale, mathematische Beschreibung

Übersetzung in ein ausführbares Modell

Textuelle Beschreibung der Semantik

Gerade für sehr ausdrucksmächtige Sprachen kann die Transformation in ein ausführbares Modell äußerst komplex werden. Daher kann die Semantikdefinition ferner über eine textuelle Beschreibung erfolgen, welche auch im BPMN-Standard für das Verhalten von Prozessdiagrammen verwendet wird. In den Beschreibungen der Semantik für die einzelnen BPMN-Elemente werden sogenannte Tokens verwendet. Ein Token wird von einem Startereignis produziert und folgt dem Kontrollfluss zur nächsten Aktivität, wo das Token dann die Bereitschaft für die Ausführung signalisiert. Nachdem diese Aktivität ausgeführt ist, traversiert das Token (oder mehrere Token) das Prozessdiagramm anhand des modellierten Kontrollflusses bis das Endereignis das Token wieder konsumiert. Auf welche Weise Token produziert, konsumiert oder weitergereicht werden wird im BPMN-Standard für einzelne BPMN-Elemente beschrieben, wodurch die Semantik eindeutig definiert ist.

2 Präziser formuliert, erlaubt DECLARE die Definition von Sprachen [139, S. 4]. Auf die Details wird hier nicht näher eingegangen.

3 gelesen in etwa: Es gilt stets ($\Box$), dass wenn Ereignis (oder Aktivität) A eintritt, tritt zu einem späteren Zeitpunkt ($\diamond$) auch Ereignis (oder Aktivität) B ein.

Aktivitätslebenszyklus

AKTIVITÄTSLEBENSZYKLUS Ähnlich den Prozessen im Prozesslebenszyklus durchlaufen auch Aktivitäten verschiedene Phasen eines Lebenszyklus. Der Aktivitätsstatus wird während der Ausführung vom Prozessausführungssystem aktualisiert. Der Aktivitätslebenszyklus kann flexibel definiert werden und je nach Bedarf detaillierter mit mehreren Stufen oder weniger detailliert mit entsprechend weniger Stufen implementiert werden. Abbildung 6 zeigt die essenziellen Stufen `inaktiv`, `bereit`, `aktiv` und `beendet` innerhalb eines möglichen Lebenszyklusmodells.

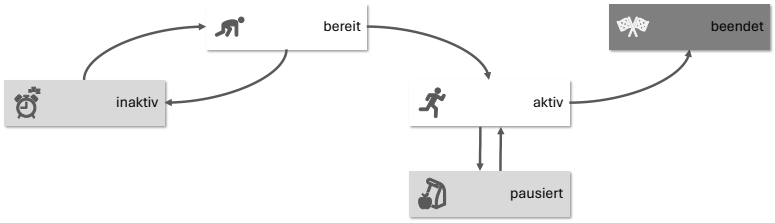

Abbildung 6: Ein Beispiel eines Aktivitätslebenszyklus

Eine fortgeschrittene Umsetzung kann weiterhin noch Phasen wie `blockiert`, `pausiert`, `fehlgeschlagen` oder `abgebrochen` integrieren ([76, Kap. 4.2.1], [135, Kap. 13.3.2]).

Implementierung der Semantik im WFMS

AUSFÜHRUNG IN EINEM SOFTWARESYSTEM (WFMS) Sollen Prozessmodelle softwaregestützt ausgeführt werden, müssen Anwendungen implementiert werden, welche in der Lage sind, die Semantik einer Prozessmodellierungssprache zu verstehen, sodass die modellierten Prozesse interpretiert werden können. Dabei kann auf die oben beschriebenen Möglichkeiten zurückgegriffen werden. Eine generische Implementierung mittels Petri-Netzen für die Ausführung von BPMN-Modellen ist wohl mit vergleichsweise wenig Aufwand verbunden, jedoch sind Petri-Netze für die Abbildung fortgeschrittener Konstrukte wie komplexe, datengetriebene Entscheidungen oder externe Ereignisse eher ungeeignet [31]. Alternativ kann die textuelle Beschreibung der Semantik als Art Lastenheft gesehen werden, worauf basierend direkt die Semantik von Modellierungssprachen implementiert werden kann. Im Falle des Camunda WFMS wird die Semantik von BPMN direkt in Java implementiert, unter anderem über eine Klasse `CoreExecution`, welche die Ausführung einer Aktivität organisiert.[4]

4 https://docs.camunda.org/javadoc/camunda-bpm-platform/7.20/org/camunda/bpm/engine/impl/core/instance/CoreExecution.html, besucht am 27.01.2024

```
1  class Task extends BPMNElement {
2      String name;
3      SequenceFlow incoming;
4      SequenceFlow outgoing;
5      boolean active;
6  }
7
8  class SequenceFlow {
9      BPMNElement sourceRef;
10     BPMNElement targetRef;
11 }
12
13 class ProcessModel {
14     List<Task> tasks;
15     List<SequenceFlow> sequenceFlows;
16 }
17
18 public void execute(Task task) {
19     if(task.active) {
20         task.active = false;
21         task.outgoing.targetRef.active = true;
22     }
23     return;
24 }
```

Quelltext 1: Rudimentäres Prozessausführungssystem in Java

Rudimentäres WFMS in Java

Quelltext 1 zeigt die Implementierung eines sehr rudimentären WFMS, worin lediglich Aktivitäten (Zeilen 1–6) und deren Sequenzfluss (Zeilen 8–11) berücksichtigt sind. In der Methode `execute` wird geprüft, ob die auszuführende Aktivität aktiv ist, also zum Beispiel, ob ein Token bei dieser Aktivität vorhanden ist (Zeile 19). Falls ja, wird der Aktivstatus auf `false` gesetzt (Zeile 20) und der Status der Zielaktivität des ausgehenden Sequenzflusses, also der nächsten auszuführenden Aktivität, auf `true` gesetzt (Zeile 21). In der Bildsprache der Tokens würde dies das Token der ausgeführten Aktivität über den Sequenzfluss zur nächsten Aktivität schieben.

Eine klar definierte Syntax und Semantik ermöglichen, dass Computerprogramme die Prozessmodelle lesen und interpretieren können. Dieser Vorgang wird auch als computergestützte oder softwaregestützte Ausführung eines Prozessmodells bezeichnet. Im folgenden Kapitel werden die Architektur und Funktionsweise von fortgeschrittenen WFMSs genauer beschrieben.

2.4 WORKFLOW-MANAGEMENT-SYSTEM

Prozessausführung im WFMS vs. modellgetriebene Softwareentwicklung

Wenn die Abarbeitung eines realen Prozesses mithilfe einer Software unterstützt werden soll, gibt es bezüglich des Softwaredesigns unterschiedliche Ansätze, etwa den kompilierenden Ansatz oder die aktive Interpretation eines Prozessmodells innerhalb eines WFMS. Beim kompilierenden Ansatz, wie zum Beispiel in [185] umgesetzt, wird nach den Prinzipien der modellgetriebenen Softwareentwicklung basierend auf einem Prozessmodell automatisiert ausführbarer Quell-

text generiert. Somit kann durch den Programmfluss die Ausführung der richtigen Aufgaben zum richtigen Zeitpunkt sichergestellt werden, was durch entsprechende Eingabemasken in der grafischen Benutzerschnittstelle unterstützt werden kann. Diese Programme sind dabei jeweils nur für den durch das ursprüngliche Prozessmodell beschriebenen Anwendungsfall entworfen, spezialisiert und optimiert und können nicht für andere Prozessmodelle wiederverwendet werden. Im Gegensatz zu Prozessausführungssystemen sind diese Programme sehr unflexibel, da beispielsweise bei Anpassungen im Prozessmodell der komplette Quelltext neu generiert und kompiliert werden muss. Demgegenüber steht die Interpretation von Prozessmodellen in einem WFMS, womit sich diese Arbeit beschäftigt. Die grundlegende Architektur wird in den folgenden Abschnitten näher erläutert.

2.4.1 *Terminologie und Definitionen*

Es existieren verschiedene Begriffe zur Beschreibung von Prozessausführungssystemen, die oft synonym verwendet werden, sich allerdings im Detail in der bereitgestellten Funktionalität unterscheiden. Der folgende Abschnitt erläutert kurz die Unterschiede und folgt dabei den Ausführungen in [39].

BPMS versus WFMS

Die oben angesprochenen Softwaresysteme, in welchen die Prozessausführungslogik implementiert ist, werden entweder als BPMS (*Business Process Management System*) oder WFMS (*Workflow-Management-System*) bezeichnet. Nach Dumas et al. [39, Kap. 9.1.2] ist ein BPMS ein System für den Entwurf, die Ausführung und Überwachung sowie die Analyse von Geschäftsprozessen und zielt damit darauf ab, alle Phasen des Prozesslebenszyklus abzudecken. Diese BPMSs sind aus den WFMSs heraus entstanden, welche sich eher auf die Modellierung und Ausführung konzentrieren. Obwohl Camunda als *Production workflow system* ebenfalls als ein BPMS klassifiziert wird [39, S. 345], wird in dieser Arbeit der Begriff WFMS verwendet, um die Fokussierung auf die Modellierungs- und vor allem auf die Ausführungsphase zu verdeutlichen.

Prozess versus Workflow

Damit stellt sich noch die Frage nach dem Unterschied zwischen einem Prozess und einem Workflow [76]. Während ein Prozess den allgemeinen Ablauf in einem Unternehmen auf konzeptioneller Ebene beschreibt, bezeichnet der Begriff Workflow die implementierte und möglicherweise automatisierte Version eines Prozesses beziehungsweise eines Prozessmodells, welches in einem WFMS interpretiert und ausgeführt werden kann [39, S. 345]. In diesem Sinne ist der Workflow im Gegensatz zum Prozess mit detaillierteren Implementierungsinformationen angereichert. Aufgrund der Ähnlichkeit der bezeichneten Artefakte werden die Begriffe Prozess und Workflow in dieser Arbeit synonym verwendet.

2.4.2 *Hauptkomponenten und ihre Aufgaben*

Referenzarchitektur

Referenzarchitekturen geben die Grundstruktur von Systemen vor und dienen als Art Blaupause, welche für bestimmte Anwendungsfälle angepasst oder erweitert werden kann. Die Autoren Grefen et al. veröffentlichten 1998 eine Referenzarchitektur für WFMSs [57], welche die wichtigsten Komponenten sowie deren internen Interaktionen und externen Schnittstellen umfasst und als Grundlage für die Entwicklung von konkreten WFMSs dienen soll. In der Referenzarchitektur sind über die Modellierung und Ausführung hinaus noch Module zur Überwachung des Prozessstatus und zur Administration enthalten. Eine erweiterte und aktualisierte Referenzarchitektur für BPMSs wird von Pourmirza et al. vorgeschlagen, bei der die Ansätze und Entwicklungen aus der akademischen Welt und kommerzielle Produkte der vergangenen Jahre mit berücksichtigt sind [141]. Der oben definierten Terminologie folgend, sind dabei auch Module für die Analyse von Prozessen mit berücksichtigt, welche in dieser Arbeit weniger von Bedeutung sind. Dieser Abschnitt definiert die für diese Arbeit essenziellen Komponenten von WFMSs und bezieht sich dabei auf die Ansätze in [39, 57, 72, 141].

Die Hauptkomponenten eines WFMS umfassen ein Modul zur Prozessmodellierung, eine Ausführungskomponente oder Ausführungsmaschine sowie die Schnittstellen zu Mitarbeitenden und zu externen Systemen.

Modellierung von Prozessen

PROZESSMODELLIERUNG Im Modul zur Prozessmodellierung kann der Prozessmodellierer ein Modell eines realen Prozesses erstellen. In Camunda ist dies über einen grafischen Editor[5] umgesetzt, welcher den Modellierer durch Vorschläge oder Hinweise unterstützt und fehlerhafte, syntaktisch inkorrekte Eingaben verhindert. Der in Camunda integrierte Modellierer bietet darüber hinaus noch Funktionalität für das Bereitstellen (englisch: *Deployment*) von Prozessmodellen (oder Workflows) in einer laufenden Camunda-Instanz.

AUSFÜHRUNGSMASCHINE Die Ausführungsmaschine (englisch: *Execution Engine*) ist einerseits für die Instanziierung der bereitgestellten Prozessmodelle verantwortlich und andererseits für die Interpretation beziehungsweise Ausführung der so erzeugten Prozessinstanzen.

Instanziierung von Prozessmodellen

Ein Prozess wird per definitionem wiederholt in einem Unternehmen ausgeführt, wobei sich die Ausführung von Fall zu Fall leicht unterscheiden kann. Im Prozessmodell werden dabei alle möglichen Varianten abgebildet. Für jeden einzelnen konkreten Vorgang wird eine neue Instanz des Modells erstellt, das heißt, das Prozessmodell wird instanziiert. Prozessinstanzen haben

5 https://camunda.com/de/download/modeler/, besucht am 27.01.2024

ein eindeutiges Identifikationsmerkmal, zum Beispiel eine Bestellnummer in einem Bestellprozess, und darüber hinaus können für jede Instanz die ausgeführten Schritte, die verarbeiteten Datenwerte oder die beteiligten Mitarbeiter gemäß den Regeln im Modell beliebig und unabhängig von früheren Instanzen variieren. Dementsprechend können unterschiedliche Prozessinstanzen auch unterschiedliche Ausführungspfade im Prozessmodell wahrnehmen. Es obliegt der Verantwortung des WFMS, mehrere Instanzen eines Prozessmodells zu verwalten, inklusive aller relevanten Datenwerte.

Ausführung von Prozessinstanzen

Die Ausführung von Prozessinstanzen umfasst im Kern die Verteilung der aktuell auszuführenden Arbeitsschritte zu sogenannten *Agenten*, welche weiter als menschliche Ressourcen oder externe Softwaresysteme klassifiziert werden können [76]. Die Ausführungsmaschine kümmert sich weiterhin um die Konsolidierung und Bereitstellung von relevanten Informationen und Daten, die zur Abarbeitung der Arbeitsschritte benötigt werden.

Bereitstellung von Arbeitslisten

ARBEITSLISTEN Über die Arbeitslisten (englisch: *Worklist*) können Mitarbeiter aktuell offene oder ihnen bereits zugewiesene abzuarbeitende Aktivitäten inspizieren. Die Ausführungsrechte offener Aktivitäten können dabei beansprucht werden (englisch: *claim*), während beanspruchte Aktivitäten unter anderem gestartet (englisch: *start*) oder abgeschlossen (englisch: *complete*) werden können. Die Verwaltung der Aktivitäten über die Arbeitslisten aktualisiert dabei den Aktivitätsstatus entsprechend dem implementierten Aktivitätslebenszyklus, welcher für die Umsetzung von Arbeitslisten zusätzliche Stufen, zum Beispiel `zugewiesen`, implementieren kann. Ähnlich zu den Arbeitslisten kann das WFMS auch entsprechende Formulare anzeigen, um etwa die für das erfolgreiche Abschließen einer Aktivität notwendigen Daten abzufragen.

Integration externer Dienste und Anwendungen

EXTERNE ANWENDUNGEN UND DIENSTE Bei der Automatisierung von Prozessen spielen externe Dienste und Anwendungen eine entscheidende Rolle. Im Prozessmodell können bestimmte Aktivitäten dahingehend konfiguriert werden, dass die Zuständigkeit für deren Abarbeitung an externe Applikationen abgegeben wird. Das WFMS ruft dann die entsprechenden Anwendungen zum richtigen Zeitpunkt auf. Es ist nicht ausgeschlossen, dass dabei auch mit weiteren externen WFMSs interagiert wird, etwa wenn ein Prozess in mehrere Subprozesse aufgeteilt ist, welche von verschiedenen WFMSs ausgeführt werden, oder auch im Rahmen der zwischenbetrieblichen Prozessausführung (Kapitel 3).

Administration und Überwachen

ADMINISTRATION UND ÜBERWACHUNG Weitere Komponenten eines WFMSs betreffen die Administration und Überwachung der Ausführung. Administrative Aufgaben umfassen unter anderem die Verwaltung von Benutzerkonten, wofür externe Benutzerverwaltungssysteme oder Verzeichnisdienste an das WFMS angebunden werden können. Die Benutzerverwaltungssysteme verwalten alle möglichen Akteure der Prozessausführung und sind für deren Authentifizierung verantwortlich. Über die Überwachungskomponente kann jeweilis der Ausführungsfortschritt, die Datenwerte, zugewiesene Mitarbeiter oder die Dauer der Ausführung von Instanzen oder einzelnen Aktivitäten geprüft werden.

2.4.3 *Vorteile beim Einsatz eines Workflow-Management-Systems*

Der Einsatz eines WFMS mit den oben beschriebenen Komponenten in einem Unternehmen bringt verschiedene Vorteile mit sich [39, Kap. 9.2].

Verringerung des Arbeitspensums

Zum einen sorgt das WFMS für eine Verringerung des Arbeitspensums von Mitarbeitern, welche nicht mehr überlegen müssen, welche Schritte als Nächstes auszuführen sind, sondern neue Aufgaben aus der Arbeitsliste entnehmen können.

Teilautomatisierung

Durch das WFMS können automatisierte Aufgaben auch ohne manuellen Eingreifen zum richtigen Zeitpunkt zur Ausführung gebracht werden.

Bereitstellung erforderlicher Daten

Des Weiteren müssen die Mitarbeiter nicht mehr manuell die relevanten Informationen aus verschiedenen Quellen zusammentragen, da das WFMS automatisch für eine Konsolidierung relevanter Informationen sorgt und die Daten aus den entsprechenden Datenbanken oder externen Applikationen automatisch bereitstellt.

Integration verschiedener Anwendungen

In diesem Sinn werden auch die heterogenen Anwendungen der IT-Systemlandschaft im Unternehmen über eine koordinierende Komponente integriert.

Prozesskonformität

Weitere entscheidende Vorteile eines WFMS sind die Einhaltung der Prozesskonformität und die transparente Überwachung der Ausführung. Getrieben durch die Arbeitslisten werden die Mitarbeiter in ihrer Arbeit geleitet, wodurch das Risiko fehlerhafter Ausführungen oder das Auslassen von Aktivitäten verringert werden kann.

Bereitstellung eines Ereignisprotokolls zur Analyse

Bei automatisierten Aufgaben sorgt das WFMS für eine konforme Abarbeitungsreihenfolge. Durch eine durchgehende Überwachung der Prozessausführung produziert das WFMS ein sogenanntes Ereignisprotokoll. Dies kann für weiterführende Analysen verwendet werden, um etwa die Zuweisungen von Aktivitäten zu Mitarbeitern zu optimieren oder Auffälligkeiten hinsichtlich der Abarbeitungsdauer zu identifizieren.

2.5 PERSPEKTIVEN EINES PROZESSMODELLS

Wichtige Perspektiven

Im bisherigen Verlauf des Kapitels sind die Teilschritte und deren zeitliche Relation als wichtigste Bausteine beziehungsweise *Perspektiven* oder *Aspekte* eines Prozessmodells beschrieben. Diese finden sich als *funktionale Perspektive* (Prozessteilschritte oder Aktivitäten) und *verhaltensorientierte Perspektive* (Kontrollfluss) im Konzept des multiperspektivischen Prozessmodells wieder [78]. Für den praktischen Einsatz in Unternehmen sind diese beiden Perspektiven oftmals nicht ausreichend und für eine praxisorientierte Ausführung werden weitere Informationen benötigt, welche den Mehrwert der Prozessausführung für die Unternehmen noch steigern. Die zusätzlichen Informationen lassen sich in weitere Perspektiven kategorisieren und ebenfalls in einem multiperspektivischen Prozessmodell abbilden [78]. Im Folgenden werden die wichtigsten Perspektiven oder Aspekte eines Prozessmodells erörtert und deren praktische Anwendbarkeit eruiert. Die Erläuterungen stehen dabei im engen Bezug zu der Modellierungssprache BPMN und zum WFMS Camunda, da diese in der restlichen Arbeit für die Modellierung beziehungsweise Ausführung der Prozesse verwendet werden. Das bedeutet, zusätzlich zur systemagnostischen Darstellung der Prozessperspektiven in der Originalliteratur liegt der Fokus hier auch auf der Anwendbarkeit und der möglichen Umsetzung der Perspektiven in aktuellen Modellierungssprachen und Ausführungssystemen. Die folgenden Kapitel, welche die Prozessperspektiven vorstellen, strukturieren sich damit in die drei Teile:

Weitere wichtige Perspektiven

- Konzeptionelle Sichtweise
- Umsetzung in BPMN
- Implementierung in Camunda

Die folgenden Perspektiven werden von Jablonski et al. in [76] und [78, Kap. 6] vorgeschlagen und lassen sich in verschiedenen Konzepten, Modellierungssprachen und Ausführungssystemen wiederfinden. Im Folgenden werden die funktionale und verhaltensorientierte Perspektive sowie die organisationale, die informationsorientierte und die operationale Perspektive genauer vorgestellt. Neben diesen bewährten und universell einsetzbaren Perspektiven kann für spezielle Domänen die Integration von weiteren Aspekten in die Prozessmodelle sinnvoll sein. Diese Aspekte umfassen unter anderem Sicherheitsbelange wie etwa Zugriffsbeschränkungen, historische Informationen aus bereits abgeschlossenen Prozessinstanzen oder die Verwendung weiterer Kontextinformationen [78, Kap. 6.6]. Die im Folgenden vorgestellten Perspektiven lassen sich inhaltlich voneinander abgrenzen, allerdings können über Referenzen auch Bezüge zwischen den Perspektiven herstellt werden.

2.5.1 *Die funktionale Perspektive*

Dieses Kapitel führt Aktivitäten und deren Typen als grundlegende Bausteine von Prozessen ein.

2.5.1.1 *Konzeptionelle Sichtweise*

Hierarchie von Prozessen

Die funktionale Perspektive definiert alle Teilschritte oder Aktivitäten (englisch: *Task*), welche innerhalb eines Prozesses zur Erreichung des vorgegebenen Ziels zu erledigen sind. Alle Teilschritte lassen sich hierarchisch in Kind- und Elternprozesse strukturieren (in [76]: *subworkflow* und *superworkflow*). Die hierarchische Strukturierung hilft dabei, die tatsächlichen Sachverhalte realitätsnah zu modellieren und dient darüber hinaus der Lesbarkeit des Modells beim Einsatz als Dokumentationsartefakt. Weiterhin kann diese Struktur unterstützend bei der Definition von Lese- und Schreibrechten eingesetzt werden, zum Beispiel, um zu entscheiden, ob Datenwerte von unterschiedlichen Hierarchiestufen aus verändert oder gelesen werden dürfen.

Aktivitätstyp

In dieser Arbeit sollen auch die unterschiedlichen Typen von Aktivitäten in der funktionalen Perspektive betrachtet werden, um im weiteren Verlauf entscheiden zu können, welche relevanten Ausführungsdetails den Aktivitäten zugeordnet werden sollen. Die Definition der Aktivitätstypen ist an den BPMN-Standard angelehnt (Kapitel 2.5.1.2) und referenziert die weiteren Prozessperspektiven. In der verhaltensorientierten Perspektive muss für einen *Business Rule Task* beispielsweise eine Entscheidungstabelle definiert werden (Kapitel 2.5.2). Die Durchführung eines *User Task*s erfordert in der organisationalen Perspektive die Zuordnung eines Mitarbeiters oder einer Abteilung aus einem Organisationsmodell, welche für die Ausführung zuständig oder verantwortlich sind (Kapitel 2.5.3). In der operationalen Perspektive wird für einen *Script Task* auszuführender Programmcode definiert (Kapitel 2.5.5).

2.5.1.2 *Umsetzung in BPMN*

Der BPMN-Standard definiert unter anderem für die oben genannten Aktivitätstypen jeweils eine spezialisierende Klasse im Metamodell der Sprache [135, Kap. 10.3.3]. Diese lassen sich an dem entsprechenden Icon in der linken oberen Ecke der Aktivität identifizieren. Die Aktivitäten werden im Folgenden in den jeweiligen Kapiteln der zugehörigen Prozessperspektiven beschrieben.

2.5.1.3 *Implementierung in Camunda*

Das Camunda WFMS implementiert die verschiedenen Aktivitätstypen und führt geeignete Funktionalitäten für deren Abarbeitung aus. So wird zum Beispiel entschieden, in welche Arbeitslisten die abzuarbeitenden User Tasks hinzugefügt werden müssen (Kapitel 2.5.3).

Weiterhin werden *Script Task*s oder *Service Task*s automatisiert vom WFMS ausgefürt, indem die konfigurierten Skripte abgearbeitet oder externe Anwendungen aufgerufen (Kapitel 2.5.5) werden.

2.5.2 *Die verhaltensorientierte Perspektive*

In der verhaltensorientierten Perspektive wird die einzuhaltende zeitliche Reihenfolge der Teilschritte beziehungsweise der Kontrollfluss (englisch: *control flow* oder *sequence flow*) spezifiziert.

2.5.2.1 *Konzeptionelle Sichtweise*

sequenzielle Abarbeitung

Im einfachsten Fall kann die sequenzielle Abarbeitung zwischen zwei Aktivitäten (oder Kindprozessen) definiert werden, wobei die Bearbeitung der abhängigen Aufgabe auf die Beendigung der abhängenden Aufgabe warten muss.

parallele Abarbeitung

Für eine beschleunigte Abarbeitung eines Prozesses kann für bestimmte Aktivitäten auch die zeitgleiche oder parallele Ausführung spezifiziert werden, zum Beispiel, wenn diese unabhängig voneinander zu bearbeiten sind. Auch komplexere Sachverhalte wie etwa eine erlaubte Parallelität oder eine geforderte Parallelität können ausgedrückt werden. Im ersten Fall sind die Teilschritte vollkommen unabhängig voneinander zu bearbeiten, während im zweiten Fall der Start einer parallelen Aktivität vor dem Beenden einer korrespondierenden Aktivität erfolgen muss, sodass diese tatsächlich zur gleichen Zeit bearbeitet werden. Dies wird durch die Einbeziehung des Aktivitätslebenszyklus möglich, indem hier eine Beziehung zwischen der verhaltensorientierten Perspektive und des aktuellen Status der referenzierten Aktivitäten hergestellt wird.

bedingte Abarbeitung

Die bedingte Abarbeitung zeigt eine interperspektivische Abhängigkeit und verbindet die verhaltensorientierte mit der informationsorientierten Perspektive, in der Datenwerte in die Prozessausführungsarchitektur eingeführt werden (*informationsorientierte Perspektive*, Kapitel 2.5.4). Diese Datenwerte beschreiben etwa die Temperatur einer Maschine bei Produktionsprozessen oder den Wert eines Formularfeldes, welches während der bisherigen Prozessausführung von einem Mitarbeiter im Rahmen eines User Tasks ausgefüllt wurde. Die Datenwerte können nun für die Steuerung der Prozessausführung verwendet werden, indem durch die Definition von Bedingungen die Auswahl der nächsten Aufgaben von diesen Werten abhängig gemacht wird.

2.5.2.2 *Umsetzung in BPMN*

Das folgende Beispiel führt die wichtigsten BPMN-Elemente bezüglich der verhaltensorientierten Perspektive ein.

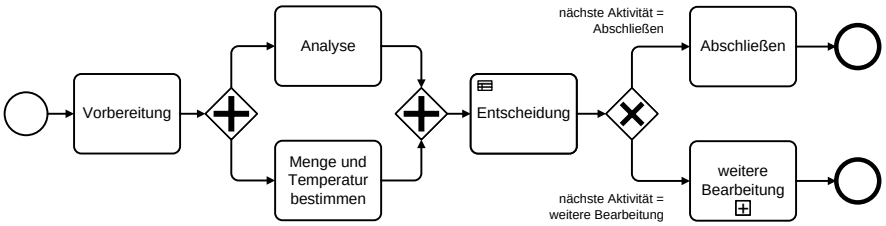

Abbildung 7: BPMN-Prozess mit einem parallelen Kontrollfluss und einem *Business Rule Task* zur Entscheidungsfindung

Beispiel: Verhaltensorientierte Perspektive in BPMN. Nachdem der Prozess in Abbildung 7 gestartet wurde, muss im Anschluss, indiziert durch den gerichteten Sequenzfluss-Pfeil, die Aktivität Vorbereitung ausgeführt werden, bevor die Aktivitäten Analyse sowie Menge und Temperatur bestimmen unabhängig voneinander und möglicherweise zeitgleich abgearbeitet werden. Die Parallelität wird in BPMN durch eine Verzweigung mit einem parallelen Gatter (englisch: *Parallel Gateway*), dargestellt durch eine Raute mit +, ausgedrückt. Im Anschluss legt der *Business Rule Task* Entscheidung fest, welcher Pfad nach dem exklusiven Gatter (englisch: *Exclusive Gateway*), dargestellt durch eine Raute mit x, ausgeführt werden soll. Die Verzweigung des Kontrollflusses durch ein *Exclusive Gateway* stellt eine bedingte Abarbeitung dar, wobei die ausgehenden Sequenzflüsse mit den zu prüfenden Bedingungen annotiert sind. Somit hängt die Ausführung der Aktivität Abschließen oder des Kindprozesses weitere Bearbeitung von der Belegung der Variablen nächste Aktivität ab. Die Variablenzuweisung erfolgt dabei regelbasiert in der Aktivität Entscheidung (Kapitel 2.5.2.3).

2.5.2.3 *Implementierung in Camunda*

Bedingungen mit (DMN)

Camunda sorgt dafür, dass die korrekten Aktivitäten zur richtigen Zeit zur Ausführung gebracht werden. Dies umfasst zum Beispiel, dass zwei Aktivitäten nach einem öffnenden parallelen Gatter unabhängig voneinander abgearbeitet werden können. Für die Spezifikation komplexerer Bedingungen, basierend auf mehreren zu verknüpfenden Datenwerten, kann ein BPMN *Business Rule Task* (Entscheidung in Abbildung 7) verwendet werden. Das Camunda WFMS bietet für die Entscheidungsfindung verschiedene Implementierungsvarianten an, zum Beispiel mit der Decision Model and Notation (DMN) [136]. Dafür wird ein Business Rule Task mit einer Tabelle verknüpft, welche die Logik zur Evaluation implementiert. Als Beispiel dient Tabelle 1, welche der Variablen nächste Aktivität den Wert weitere Bearbeitung zuordnet, wenn der Wert der Variablen Menge größer als 10.000 ist und der Wert der Variablen Temperatur kleiner als 20. Entspre-

chend werden die restlichen Bedingungen geprüft und der Wert der Variablen nächste Aktivität gesetzt.

Wenn	Und	Dann
Menge	Temperatur	nächste Aktivität
> 10.000	< 20	weitere Bearbeitung
⩽ 10.000	< 15	Abschließen

Tabelle 1: DMN-Tabelle zur Entscheidungsfindung

2.5.3 *Die organisationale Perspektive*

Ein großer Vorteil der systemgestützten Prozessausführung zeigt sich mit der Integration der organisationalen Perspektive, welche ein Organisationsmodell eines Unternehmens mit einem Prozessmodell verknüpft, mit dem Ziel, geeignete Personen für die Abarbeitung von Teilaufgaben (User Tasks) auszuwählen. Bevor die Details erläutert werden, skizziert der folgende Abschnitt kurz die betriebswirtschaftlichen Hintergründe.

BETRIEBSWIRTSCHAFTLICHER HINTERGRUND Die Betriebswirtschaftslehre unterscheidet bei Unternehmen die sogenannte Aufbauorganisation [56] von der Ablauforganisation [55]. In einem Unternehmen liefert die Aufbauorganisation die hierarchische Strukturierung des gesamten Personals und sorgt dabei für die Definition von Stellen, Abteilungen oder Rollen, welche fortan unter dem Begriff der *organisationalen Einheit* oder *organisationalen Entität* verallgemeinert geführt werden. Die organisationalen Entitäten zeichnen sich durch ihre jeweiligen Kompetenzen aus und regeln die Zuständigkeiten für Aufgabenbereiche innerhalb des Unternehmens.

Aufbauorganisation

Die Ablauforganisation hingegen spezifiziert, wie die organisationalen Einheiten der statischen Unternehmensstruktur in die Geschäftsprozesse zu integrieren sind. Aufbauorganisation und Ablauforganisation werden vorerst getrennt voneinander betrachtet und in unterschiedlichen Modellen verwaltet, nämlich den Organisationsmodellen und Prozessmodellen. Die Geschäftsprozesse, welche den Weg zur Erreichung der Unternehmensziele definieren, sind dabei ohne eine verfügbare Organisationsstruktur nicht realisierbar [154, S. 27]. Der Forschungsschwerpunkt dieser Arbeit ist mit der softwaregestützten Prozessausführung im Bereich der Ablauforganisation anzusiedeln. Dieses Kapitel beschreibt dabei die Verzahnung zwischen der Prozessausführung und der statischen Unternehmensstruktur aus der Sicht des Geschäftsprozessmanagements.

Ablauforganisation

2.5.3.1 *Konzeptionelle Sichtweise*

Organigramm

DEFINITION VON ORGANISATIONSSTRUKTUREN Das Ziel der organisationalen Perspektive ist es, organisationale Entitäten eines Organigramms aus dem Prozessmodell heraus zu referenzieren, um Verantwortlichkeiten von Teilaufgaben im Prozessmodell zu definieren. Ein Organisationsmodell oder Organigramm stellt dabei generell ein Modell der Unternehmensstruktur dar. Das Organisationsmodell modelliert alle Mitarbeiter eines Unternehmens beziehungsweise alle möglichen beteiligten Personen einer Prozessausführung inklusive deren hierarchische Strukturierung und beliebig weiteren Attributen. Dabei wird ähnlich zum Prozessmodell nach der allgemeinen Modelltheorie [161] zweckdienlich von irrelevanten Details abstrahiert. Alle Einzelpersonen, hierarchische Gruppierungen von Personen, zum Beispiel Abteilungen in Unternehmen, Lehrstühle an einer Universität oder Trupps, Gruppen, Züge und so weiter bei der Bundeswehr[6], können fortan aus dem Prozessmodell heraus referenziert werden. Darüber kann festgelegt werden, welche Personen oder Personengruppen für die Ausführung bestimmter Teilschritte im Prozess infrage kommen. Ebenso können für diesen Zweck auch weitere organisationale Entitäten, Attribute oder Rollenbezeichnungen wie Abteilungsleiter, Professor, Sekretärin oder Zugführer referenziert werden. Während der Prozessausführung kann somit stets eine Menge der geeigneten Personen für bestimmte Teilaufgaben anhand des Organigramms abgeleitet werden.

praxisnahe, intuitive Modellierung

Die genannten Beispiele zeigen die hohen Anforderungen bezüglich einer flexiblen Gestaltung der Organisationsmodelle, da verschiedene Institutionen ihrerseits die Gruppierungen von Mitarbeitenden oder Personen unterschiedlich benennen. Zwar kann in einem Organigramm einer Universität die Gliederung in Lehrstühle auch über ein Schlüsselwort *Gruppe* erfolgen, worin Mitarbeitern bestimmte *Rollen* zugewiesen sind, jedoch sollte nach Bussler et al. eine derart abstrakte Terminologie nur auf Metamodellebene verwendet werden [18]. Stattdessen sind auf Modellebene aussagekräftigere Bezeichnungen für Gruppen, wie etwa *Lehrstuhl*, *Institut* oder *Fakultät*, vorzuziehen. Dies erlaubt eine treffendere Modellbildung, wie am folgenden Anwendungsszenario im Universitätskontext gezeigt.

Beispiel: Organisationsmodellierung Lehrstuhl AI4. Für den Anwendungsfall Universität soll basierend auf dem Metamodell von Bussler et al. [18, Kap. 6.3.2] ein Organisationsmodell erstellt werden, dessen Instanziierung eine reale Organisationsstruktur abbildet. Abbildung 8 zeigt den relevanten Ausschnitt des Metamodells (links), das erstellte Organisationsmodell, welches universitäre Strukturen im

6 https://www.bundeswehr.de/de/ueber-die-bundeswehr/zahlen-daten-fakten/militaerische-einheiten-bundeswehr, besucht am 14.11.2023

Allgemeinen abbildet (mittig) sowie ausschnittsweise die Instanziierung des Modells für die Universität Bayreuth (rechts).

Der Organisationsstrukturtyp (UNI) ist zur Vereinfachung im weiteren Verlauf ausgelassen. Agenttypen (Mitarbeiter, Professor) referenzieren existierende Entitäten in der realen Welt, während Nichtagenttypen abstrakte Dinge repräsentieren, welche nicht direkt Aufgaben abarbeiten können (Lehrstuhl, Rolle). Im Beispiel werden zwei Mitarbeiter (KH, CS) sowie ein Professor (SJ) als Agenttypen modelliert sowie ein Lehrstuhl (AI4) und eine Rolle (Sekretärin) als Nichtagenttypen.

Weiterhin sind drei Organisationsbeziehungstypen definiert, angestellt (ANG), Sekretariat (SEK) und HeadOfChair (HOC), wobei die zu referenzierenden Entitäten in Klammern angegeben sind. Beispielsweise wird der Beziehungstyp angestellt mit einer Instanz eines Mitarbeiters (KH oder CS) und eines Lehrstuhls (AI4) instanziiert.

Über Instanzattribute können die organisationalen Entitäten näher beschrieben werden. In der Abbildung 8 sind mit Name, E-Mail und Raum drei Instanzattribute für Mitarbeiter definiert sowie ein Instanzattribut Name für Lehrstuhl. Im instanziierten Modell ist der Instanz AI4 vom Nichtagenttyp Lehrstuhl über die Instanziierung des Instanzattributs Name der Name DBIS&IS zugeordnet.

Zuletzt wird mit HAT_ROLLE eine Organisationsinstanzoperation definiert, welche alle Instanzen des Agenttyps Mitarbeiter zurückgibt, die über einen Organisationsbeziehungstyp, wie zum Beispiel SEK, mit einer entsprechenden Rolle verknüpft sind. HAT_ROLLE kann dementsprechend so implementiert werden, dass der Aufruf HAT_ROLLE(Sekretärin) eine Liste an Mitarbeitern zurückgibt, welchen über SEK die Rolle Sekretärin zugeordnet ist, im Beispiel, bestehend aus dem einzigen Element KH.

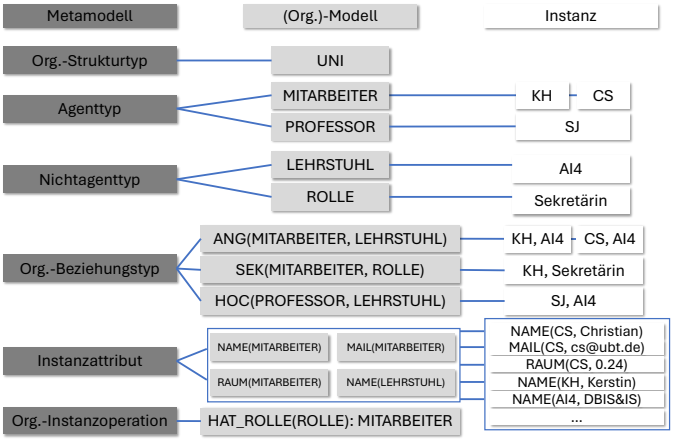

Abbildung 8: Ausschnitt eines Organisationsmodells einer Universität unter Verwendung des Metamodells nach Bussler et al. [18]

Flexibilität in der Organisationsmodellierung

An dem obigen Beispiel ist gezeigt, dass durch das allgemeine Metaschema als Grundlage, die Organisationsmodellierung intuitiv und unter Verwendung der auch im Sprachgebrauch üblichen Begriffe möglich ist [18, Kap. 6.3]. Weiterhin stellt sich die Flexibilität heraus, entweder einen Nichtagenttyp Rolle zu definieren, dessen Instanziierung Sekretärin über einen Organisationsbeziehungstyp an eine Instanziierung des Agenttyps Mitarbeiter zugewiesen werden kann, oder alternativ direkt einen Agenttyp Professor zu spezifizieren. Über die Organisationsinstanzoperationen, zum Beispiel HAT_ROLLE, ist darüber hinaus ebenfalls eine flexible Definition von Operationen auf Entitäten und deren Beziehungen möglich. Für weiterführende Konzepte, wie zum Beispiel Konsistenzregeln, sei auf die Originalliteratur verwiesen [18].

flexible Definition von Zuweisungsregeln

ZUWEISUNGSREGELN Über Zuweisungsregeln werden zu einer gegebenen Prozessinstanz und einer Prozessaktion alle organisationalen Entitäten aus dem Organisationsmodell abgeleitet, die potenziell zur Ausführung der Prozessaktion infrage kommen [18, Kap. 7.2.1]. Dies ermöglicht unter anderem die Bereitstellung von personalisierten Arbeitslisten für die Prozessakteure, wobei die Zuweisungsregeln auf Basis von Daten, welche während der Prozessausführung verarbeitet werden (Kapitel 2.5.4), dem Organisationsmodell oder historischen Daten aus dem bisherigen Ereignisprotokoll definiert werden können. Bussler et al. [18, Kap.7.2] schlagen ähnlich zur Organisationsmodellierung ein Metaschema zur flexiblen Modellierung von Zuweisungsregeln vor sowie darüber hinaus noch weitere Metaschemata, zum Beispiel zur Auflösung möglicher Konflikte während der Evaluation der Zuweisungsregeln. Diese Arbeit verzichtet in den weiteren Diskussionen auf diese flexible Möglichkeit der Definition von Zuweisungsregeln, nicht, um die Irrelevanz der vorgeschlagenen Konzepte zum Ausdruck zu bringen, sondern um den Fokus auf die Prozessausführungskomponente als Kernthema dieser Arbeit zu lenken.

Im Gegensatz zur flexiblen Integration der organisationalen Perspektive nach Bussler et al. [18], stellen BPMN und Camunda weniger fortgeschrittene Werkzeuge zur Verfügung (Kapitel 2.5.3.2 und Kapitel 2.5.3.3). Aus diesem Grund erweitert Kapitel 6 BPMN und Camunda in dieser Hinsicht. Die dort angenommene Modellierung und Implementierung der organisationalen Perspektive wird in Kapitel 2.5.3.4 vorgestellt. Dabei erfolgt die Modellierung über BPMN Annotationen und die Zuweisungsregeln werden über eine implementierungstechnische Umsetzung realisiert.

2.5.3.2 *Umsetzung in BPMN*

Pools und Lanes

Nach der offiziellen Spezifikation unterstützt BPMN ausschließlich Modellierungskonzepte bezüglich Geschäftsprozessen, während die Definition von Organisationsmodellen und Ressourcen explizit ausgeschlossen ist [135, Kap. 7.2]. Für eine Analyse, inwiefern Bedingungen hinsichtlich der organisationalen Perspektive dennoch in BPMN umgesetzt werden können, sind drei Aspekte zu betrachten: die Unterstützung im Rahmen der grafischen Notation, die Umsetzung im Metamodell und die Verwendung in WFMSs wie etwa Camunda. Als grafische Elemente stellt BPMN sogenannte *Pools* und *Lanes* zur Verfügung, welche allerdings Schwächen aufweisen. Wird bei *Pools* eine mögliche semantische Bedeutung als Repräsentation eines Teilnehmers einer Kollaboration noch angedeutet [135, S. 22], ist die Bedeutung einer *Lane* nicht definiert, sondern vom Prozessmodellierer festzulegen [135, S. 305], was eine allgemeingültige Implementierung erschwert. Da das in dieser Arbeit verwendete WFMS Camunda die *Pool*- und *Lane*-Elemente nicht unterstützt und stattdessen herstellerspezifische Erweiterungen verwendet, wird hier auf eine tiefere Analyse von BPMN und inwieweit die organisationale Perspektive abseits der grafischen Visualisierung unterstützt verzichtet. Die grafischen Darstellungen für Pools und Lanes werden in Kapitel 3 eingeführt.

2.5.3.3 *Implementierung in Camunda*

Umsetzung der organisationalen Perspektive in Camunda

Camunda verwendet BPMN zur Definition von Geschäftsprozessen, wobei *Pools* und *Lanes* nicht interpretiert werden. Stattdessen können spezielle Erweiterungen verwendet werden, um die organisationalen Verantwortlichkeiten von User Tasks auszudrücken, was hier an einem Beispiel gezeigt werden soll.[7] Im folgenden Ausschnitt eines in XML dargestellten BPMN-Modells wird der Benutzer, der den Prozess über das Startereignis instanziiert hat, über die Erweiterung `camunda:initiator` an die Prozessvariable `starter` gebunden. Im `camunda:assignee`-Attribut des darauffolgenden User Tasks wird der Benutzer dann als Verantwortlicher dieser Aktivität definiert.

```
<startEvent ... camunda:initiator="starter" />
<userTask ... camunda:assignee="${ starter }"/>
```

In einer ähnlichen Weise können darüber auch explizit einzelne Benutzer oder Benutzergruppen als mögliche Verantwortliche einem User Task zugewiesen werden. Für diese Benutzer stehen die Aktivitäten dann in der personalisierten Arbeitsliste zum Beanspruchen

7 https://docs.camunda.org/manual/7.20/reference/bpmn20/tasks/user-task/#user-assignment, besucht am 28.01.2024

bereit. Dabei verwaltet Camunda die Benutzer nicht über ein externes Organisationsmodell. Stattdessen werden diese manuell im Benutzerverwaltungssystem angelegt oder über einen externen Verzeichnisdienst wie *LDAP* eingebunden. Damit sind manuell implementierte Zuweisungsregeln nur schwer umsetzbar, vor allem, wenn diese über das Prozessmodell spezifiziert werden sollen. Da dies jedoch in Kapitel 6 als Anforderung definiert wird, wird die organisationale Perspektive in dieser Arbeit in einer externen Erweiterung umgesetzt (Kapitel 2.5.3.4).

2.5.3.4 *Modellierung und Implementierung in dieser Arbeit*

Umsetzung der organisationalen Perspektive in dieser Arbeit

Die Umsetzung der organisationalen Perspektive in dieser Arbeit erfolgt mithilfe einer WFMS-externen OrgEngine. Ein WFMS kommuniziert dabei über eine Schnittstelle mit der OrgEngine, welche die Zuweisungsregeln implementiert und evaluiert. Im Folgenden wird die Modellierung und Implementierung mit der OrgEngine beschrieben.

Organisationsmodell

MODELLIERUNG FÜR DIE ORGENGINE Die OrgEngine abstrahiert von konkreten Sprachen zur Organisationsmodellierung. Stattdessen wird angenommen, dass ein beliebiges Organisationsmodell auf ein tabellarisches Implementierungsmodell abgebildet werden kann. Das Implementierungsmodell ordnet dabei jedem möglichen Prozessakteur eine Menge an Schlüssel-Wert-Paaren als Attribute zu. Das Organisationsmodell in Abbildung 8 könnte beispielsweise in das in Tabelle 2 dargestellte zugehörige Implementierungsmodell überführt werden.

Agent	Attribute		
	Lehrstuhl	Rolle	Gruppe
CS	AI4		
KH	AI4	Sekretärin	
SJ	AI4	Professor	

Tabelle 2: Implementierungsmodell der organisationalen Struktur

Referenzierung des Organisationsmodells

Zur Definition organisationaler Bedingungen werden im BPMN-Prozessmodell die Elemente des Organisationsmodells (Implementierungsmodell) in Funktionen referenziert und über BPMN-Annotationen an User Tasks angehängt. Dies wird am folgenden Beispiel verdeutlicht.

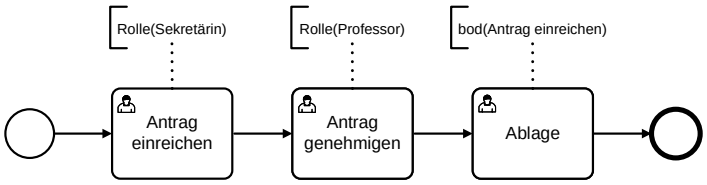

Abbildung 9: BPMN-Prozess mit organisationalen Bedingungen in den Annotationen

Beispiel: Organisationale Perspektive in BPMN mit der OrgEngine. Im Prozessmodell in Abbildung 9 sind die organisationalen Bedingungen in den Annotationen der Aktivitäten jeweils als Funktion mit Argumenten spezifiziert. Bei der Aktivität Antrag einreichen besagt die Funktion Rolle, welche mit dem Argument Sekretärin aufgerufen wird, dass diese Aktivität nur von Agenten ausgeführt werden kann, welchen im Organisationsmodell für den Schlüssel Rolle der Wert Sekretärin zugewiesen ist. Die zweite Aktivität Antrag genehmigen muss von einem Agenten mit Rolle Professor ausgeführt werden. Die Aktivität Ablage ist mit der Funktion bod (*binding of duties*) annotiert, welche besagt, dass Ablage vom gleichen Agenten ausgeführt werden muss, welcher auch die als Argument spezifizierte Aktivität (hier: Antrag einreichen) ausgeführt hat.

IMPLEMENTIERUNG IN DER ORGENGINE Quelltext 2 zeigt ausschnittsweise die Implementierung der OrgEngine. Während der Prozessausführung ruft das WFMS die Methode `evaluate` auf (Zeile 10), um für eine bestimmte Aktion die Konformität hinsichtlich der organisationalen Perspektive zu überprüfen. Dabei werden alle potenziell relevanten Daten, nämlich das Prozess- und Organisationsmodell, die Ausführungshistorie und die Aktivität selbst übergeben. Anschließend wird über den Namen, welcher in der Annotation der Aktivität spezifiziert ist, die entsprechende Funktion, zum Beispiel Rolle oder bod, gesucht (Zeile 11). Die Funktionen können wie hier etwa als `java.util.function.BiFunction` realisiert sein. Quelltext 2 zeigt in den Zeilen 3–8 die Implementierung der bod-Funktion, welche in Zeile 12 aufgerufen wird. Die Funktion konsultiert die Ausführungshistorie (`log`) und extrahiert den Agenten, welcher die in der *bod*-Funktion als Argument spezifizierte Aktivität ausgeführt hat (Zeile 5). Anschließend werden der extrahierte Agent und der im aktuellen `task` spezifizierte Agent auf Gleichheit überprüft (Zeile 6). Schließlich gibt die `evaluate`-Funktion das Ergebnis in Zeile 12 an das WFMS zurück.

Weitere Funktionen können auf diese Weise in der OrgEngine implementiert und anschließend im Prozessmodell verwendet werden, sodass eine flexible Evaluation von Zuweisungsregeln unter der Ver-

```
1  public class OrgEngine {
2
3      BiFunction<...> bod = new BiFunction<...>() {
4          boolean apply(arguments, orgModel, log, task) {
5              Agent agentToBe = log.getTask(arguments[o]).agent;
6              return task.agent == agentToBe;
7          }
8      };
9
10     public boolean evaluate(BPMNModel bpmnModel, String orgModel, EventLog log, Task
           task) {
11         BiFunction function = getFunction(task.annotation);
12         return function.apply(task.annotation.arguments, orgModel, log, task);
13     }
14 }
```

Quelltext 2: OrgEngine: Ein System zur Auswertung von Regeln im Organigramm

wendung des Prozess- und Organisationsmodells sowie den relevanten Prozessdaten erfolgen kann.

Limitationen des Implementierungsmodells

LIMITATIONEN Im Gegensatz zum Organisationsmodell nach Bussler et al. [18] erlaubt das Implementierungsmodell keine fortgeschrittenen Beziehungen. Das Organisationsmodell in Abbildung 8 kann hingegen um eine Organisationsinstanzoperation IST_SEKRETÄRIN_VON erweitert werden. Darüber könnte die Zuständigkeit der Sekretärin KH für den Mitarbeiter CS ausgedrückt werden, da den beiden Mitarbeitern über den Organisationsbeziehungstyp angestellt die gleiche Instanz des Nichtagenttyps LEHRSTUHL, nämlich AI4, zugeordnet ist. Anstatt diese Art von Sachverhalt im Organisationsmodell zu spezifizieren, werden derartige Regeln in der OrgEngine manuell implementiert. Das obige Beispiel kann zum Beispiel über die folgende SQL-Operation basierend auf der in Tabelle 2 angedeuteten Datenbank ausgedrückt oder in der OrgEngine entsprechend programmatisch, etwa in Java, umgesetzt werden.

```
SELECT o2.Agent AS Sekretärin
FROM Org o1 JOIN Org o2 on o1.Lehrstuhl = o2.Lehrstuhl
WHERE o1.Agent = 'CS' AND o2.Rolle = 'Sekretärin';
```

2.5.4 *Die informationsorientierte Perspektive*

Daten und Dokumente sind für den Betrieb und den Erfolg von Unternehmen von zentraler Bedeutung und sind auch bei der Ausführung von Geschäftsprozessen ein entscheidender Faktor. Dieser Abschnitt gibt einen kurzen Einblick in das Datenmanagement von Unternehmen und führt dann in die Verwendung der Daten bei der Prozessausführung ein.

dispositive und operative Daten

DATENMANAGEMENT IN UNTERNEHMEN Aus Sicht der Betriebswirtschaftslehre lassen sich alle Daten im Unternehmen hinsichtlich eines fortgeschrittenen Datenmanagements kategorisieren, zum Beispiel in *dispositive* und *operative Daten* [9, S. 15–16]: Während *dispositive* oder *entscheidungsorientierte Daten* auf Managementebene für strategische Entscheidungen genutzt werden, beeinflussen die *operativen Daten* direkt das operative Geschäft und damit auch die Ausführung von Geschäftsprozessen. Operative Daten werden in *OLTP*-Systemen *(Online-Transaction-Processsing)* wie relationalen Datenbanken im täglichen Betrieb erzeugt und verarbeitet, wohingegen dispositive Daten in *OLAP*-Systemen *(Online Analytical Processing)* für komplexere Analysen gehalten werden und vom laufenden Betrieb entkoppelt sind. Für die informationsorientierte Perspektive sind somit hauptsächlich die operativen Daten von Interesse.

Formate und Quellen von Daten

Daten können in unterschiedlichen Formen oder Formaten vorliegen und dabei aus unterschiedlichen Datenquellen stammen. Über die oben angesprochenen relationalen Datenbanken kann zum Beispiel eine bestimmte Produktnummer aus einem Produktkatalog abgerufen werden. Eine Auftragsbestätigung könnte beispielsweise als Datei im Office Open XML-Format (.docx) oder PDF-Format (.pdf) über ein angebundenes Dokumentenmanagementsystem bereitgestellt oder dort erzeugt werden [39, S. 4]. Daten aus kontinuierlichen Datenströmen (englisch: *streaming data*) wie Temperaturdaten von Produktionsmaschinen könnten wiederum über IoT-Sensoren integriert werden [153]. Die informationsorientierte Perspektive integriert dabei Daten unterschiedlicher Form und Herkunft in die Prozessausführung.

2.5.4.1 *Konzeptionelle Sichtweise*

Verschmelzung von Daten- und Prozesswelt

DATEN UND PROZESSE: REFERENZARCHITEKTUR Für die Integration der informationsorientierten Perspektive in die Prozessausführung ist es obligatorisch, dass das WFMS auf die benötigten Daten oder Dokumente zugreifen kann. Einen Vorschlag für die Verschmelzung der Datenwelt und der Prozesswelt liefert die Workflow Management Coalition in einer Referenzarchitektur für WFMSs, in der neben den wichtigsten Komponenten von WFMSs auch die Datenperspektive berücksichtigt wird [57]. Die Referenzarchitektur unterscheidet dabei zwischen *Workflow Control Data*, *Workflow Relevant Data (Case Data)* und *Workflow Application Data* [77, Kap. 2.6].

Prozesskontrolldaten

WORKFLOW CONTROL DATA Die *Workflow Control Data* (deutsch: Prozesskontrolldaten) bezeichnen WFMS-interne Daten, welche den reibungslosen Betrieb der Prozessausführung aufrechterhalten, etwa den aktuellen Status oder Fortschritt von einzelnen Prozessinstanzen (Ereignisprotokoll).

Prozessvariablen

WORKFLOW RELEVANT DATA (CASE DATA) Basierend auf den *Work-*

flow Relevant Data (deutsch: Prozessvariablen) trifft das WFMS Entscheidungen, welche Aktivitäten als Nächstes in die jeweiligen Arbeitslisten eingereiht werden sollen. Diese Daten werden vom WFMS verwaltet und können auch von externen Anwendungen (englisch: *Workflow Application Programs*) über eine Schnittstelle aktualisiert werden.

Anwendungsdaten

WORKFLOW APPLICATION DATA Mit den *Workflow Application Data* (deutsch: externe Anwendungsdaten) werden hingegen Daten bezeichnet, welche ausschließlich von externen Anwendungen verwaltet werden.

Aus Sicht der informationsorientierten Perspektive sind dabei die Prozessvariablen am relevantesten. Der folgende Abschnitt betrachtet deren Integration näher.

PROZESSVARIABLEN Damit das WFMS externe Anwendungsdaten für die Prozesssteuerung verwenden kann, müssen diese als Prozessvariable im Geltungsbereich des WFMS verfügbar gemacht werden. Abbildung 10 zeigt die Datenintegration aus verschiedenen externen Applikationen in die Prozessausführung. Prozessvariablen (*Workflow Relevant Data*) sind darin rot dargestellt.

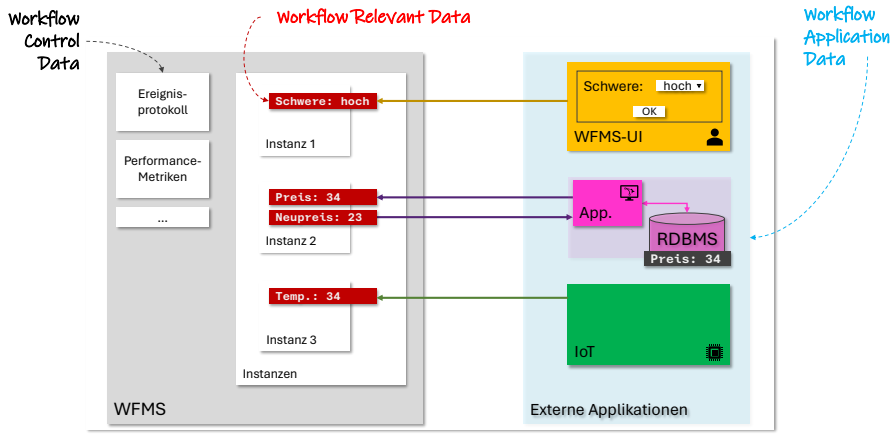

Abbildung 10: Datenintegration in die WFMS-gestützte Prozessausführung

Benutzerdefinierte Prozessvariablen

Für die Integration kann das WFMS direkt über die grafische Benutzeroberfläche ein externes Programm zur Verfügung stellen. Darüber können Daten auf Basis einer Benutzereingabe als Prozessvariable erzeugt werden. In Abbildung 10 oben rechts (orange) muss zum Beispiel ein Mitarbeiter die Schwere eines Programmfehlers einschätzen und erzeugt über die Benutzerschnittstelle im Rahmen dieser Aktivität eine Prozessvariable. Dieses Datum kann fortan die Prozesssteuerung beeinflussen. Oftmals werden auch über die Prozessausführung Daten von externen Anwendungen aktualisiert (Abbildung 10, mittig in pink). Dafür wird zuerst die entsprechende Information als Prozessvariable in das WFMS geladen. Nun kann über die Prozessausfüh-

Externe Anwendungsdaten

rung der Wert entweder benutzergesteuert oder über eine automatisierte Kalkulation verändert werden. In Abbildung 10 wird basierend auf dem ursprünglichen Preis (34) ein Neupreis von 23 berechnet. Da statt einer Referenz lediglich eine Kopie des Preis-Datums erstellt wird, ist der Neupreis wiederum ausschließlich in der Prozessvariable verfügbar. Zur Aktualisierung in der externen Anwendung muss eine dedizierte, möglicherweise automatisierte Aufgabe im Prozessmodell eingefügt werden.

Die wichtigsten Facetten der Unterstützung und Integration der informationsorientierten Perspektive hinsichtlich der Modellierung in BPMN und der Implementierung im Camunda WFMS werden im Folgenden exemplarisch gezeigt.

2.5.4.2 *Umsetzung in BPMN*

Gegeben sei der folgende Beispielprozess zur Temperaturregelung einer Produktionsanlage, welcher in Abbildung 11 als Prozessmodell in BPMN dargestellt ist.

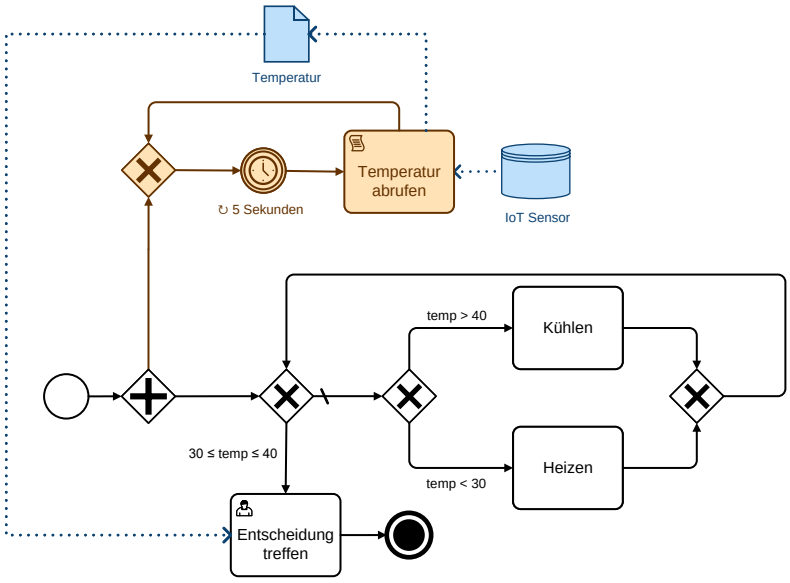

Abbildung 11: BPMN-Modell eines Prozesses zur Temperatursteuerung mit blau dargestellten *Data Objects* und *Data Stores*

Beispiel: Prozess zur Temperaturregelung. Über einen Prozess soll die Temperaturregelung einer Produktionsanlage automatisiert werden: Wenn die Temperatur, die über einen IoT-Sensor gemessen wird, außerhalb eines Bereichs zwischen 30°C und 40°C liegt, dann werden entsprechende Heiz- beziehungsweise Kühlaktivitäten ausgeführt. Nachdem die Temperatur den Vorgaben gemäß eingestellt ist, muss ein Mitarbeiter über einen User Task eine Entscheidung treffen.

Modellierungsaspekte

Im Prozessmodell in Abbildung 11 spaltet sich der Prozess zu Beginn in zwei parallele Ausführungspfade. Der obere Pfad mit orangen Modellierungselementen führt alle fünf Sekunden den Script Task Temperatur abrufen (Kapitel 2.5.5) aus, was durch das vorgeschaltete *Timer Event* [135, S. 250] erreicht wird. Die BPMN bietet in der informationsorientierten Perspektive verschiedene Elemente an [135, Kap. 10.4], unter anderem ein *Data Object* (deutsch: Datenobjekt) und einen *Data Store* (deutsch: Datenspeicher), welcher externe Informationsquellen beschreibt [135, S. 207]. Diese Elemente sind im Prozess in Abbildung 11 blau dargestellt und werden zur Modellierung der Prozessvariablen Temperatur (Datenobjekt) und der Datenquelle IoT-Sensor (Datenspeicher) verwendet. Im weiteren Verlauf beeinflusst dann die Temperatur als Datum in der informationsorientierten Perspektive den Kontrollfluss in der verhaltensorientierten Perspektive.

Implementierungsaspekte

Die Implementierung der BPMN-Elemente wird in der grafischen Darstellung nicht angezeigt und kann stattdessen in der textuellen Repräsentation in Abbildung 12 überprüft werden. Darin wird für das Timer Event der Wert `R/PT5S` im ISO 8601-Format angegeben[8] und außerdem wird ein Skript für den Script Task im JavaScript-Format bereitgestellt. Hier kann eine Verbindung zu externen Datenquellen aufgebaut werden, um die erforderlichen Daten abzurufen.

```
<bpmn:intermediateCatchEvent name=" 5 Sekunden">
    <bpmn:timerEventDefinition>
        <bpmn:timeCycle xsi:type="bpmn:tFormalExpression">
            R/PT5S
        </bpmn:timeCycle>
    </bpmn:timerEventDefinition>
</bpmn:intermediateCatchEvent>
<bpmn:scriptTask name="Temp. abrufen" scriptFormat="javascript">
    <bpmn:script>// Temperatur abrufen ...</bpmn:script>
</bpmn:scriptTask>
<bpmn:dataStoreReference name="IoT Sensor" />
<bpmn:dataObjectReference name="Temperatur" dataObjectRef="0y7"/>
<bpmn:dataObject id="0y7" />
```

Abbildung 12: Prozess zur Temperaturüberprüfung für die Dokumentation mit BPMN

Es ist allerdings zu bemerken, dass die BPMN ähnlich zur organisationalen Perspektive hinsichtlich der späteren Implementierungsphase keine fortgeschrittene Unterstützung für die Modellierung der Datenperspektive bietet [135, Kap. 2.10]. Dies zeigt sich anhand der Modellierungselemente für Datenobjekte und -speicher, welche keine Implementierungsinformationen beinhalten und damit reinen Dokumentationszweck erfüllen.

8 https://docs.camunda.io/docs/components/modeler/bpmn/timer-events/#time-duration, besucht am 18.11.2023

Risiken und Nebenwirkungen

Zusammengefasst reflektiert das BPMN-Modell, dass die Temperatur vor deren Verwendung zuerst explizit in der Prozesswelt als Datum eingeführt werden muss. Im Prozess in Abbildung 12 ist die Temperaturabfrage als nebenläufiger Kontrollfluss modelliert, welcher die Prozessvariablen mit den externen Sensordaten synchronisiert. Bei dieser Variante ist die implementierungstechnische Umsetzung des Datenabrufs explizit im Prozessmodell dargestellt. Dies kann bezogen auf den Anwendungsfall sinnvoll sein, während man in anderen Anwendungsfällen auf die Implementierungsdetails im Prozessmodell verzichten möchte. Ein weiterer Nachteil ist, dass auch explizit für die Beendigung der Synchronisierungsaktivität gesorgt werden muss. Im Beispiel wird dies durch ein *Terminate End Event* [135, S. 247] unterstützt, welches im Gegensatz zum *None End Event* [135, S. 246], welches unten bei der Implementierung in Camunda verwendet wird, beim Erreichen alle laufenden Aktivitäten und Kontrollflüsse abbricht. Dies führt jedoch möglicherweise zu ungewünschten Nebenwirkungen und ist keine allgemeingültige Lösung für alle Anwendungsfälle.

2.5.4.3 *Implementierung in Camunda*

Auch wenn sich einige Optionen zur Modellierung von Daten in BPMN bieten, werden diese wieder nicht bei der Implementierung im Camunda WFMS berücksichtigt. In der Dokumentation von Camunda wird auf die Vorteile der so erreichten Flexibilität hingewiesen.[9] Weiterhin wird vorgeschlagen, die Datenobjekte auch zu Dokumentationszwecken nur begrenzt einzusetzen.[10]

Rückruffunktion am exklusiven Gatter

Abbildung 13 zeigt ein in Camunda implementiertes Modell für den Prozess zur Temperatursteuerung. Camunda bietet für die Integration der Prozessvariablen die Möglichkeit, sogenannte *Execution listeners* als eine Art Rückruffunktion an BPMN-Elemente anzuhängen. Die Rückruffunktionen können über JavaScript-Programme definiert und je nach Konfiguration vor oder nach der Abarbeitung der entsprechenden Elemente ausgeführt werden. Im Prozessmodell in Abbildung 13 ist für das blau gefärbte exklusive Gatter eine Rückruffunktion definiert, welche den Sensorwert des externen Temperaturfühlers als Prozessvariable speichert. Damit ist sichergestellt, dass die Entscheidung des Gatters stets auf dem aktuellsten Temperaturwert getroffen wird.

9 https://docs.camunda.io/docs/components/best-practices/development/handling-data-in-processes/, besucht am 16.11.2023

10 https://docs.camunda.io/docs/components/best-practices/modeling/creating-readable-process-models/#avoiding-excessive-usage-of-data-objects, besucht am 16.11.2023

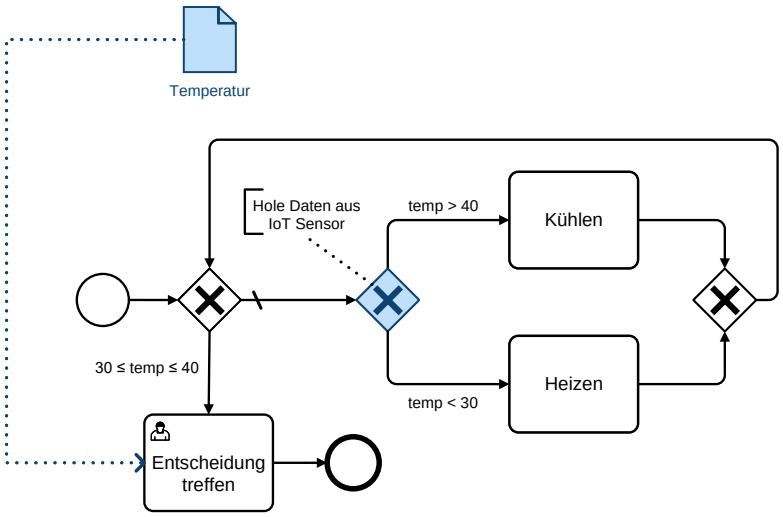

Abbildung 13: Modellierung und Implementierung eines Prozesses zur Temperaturüberprüfung mit dem WFMS Camunda

Risiken und Nebenwirkungen

Folgende Punkte sind bei der Implementierung in Camunda zu beobachten. Bei mehrmaliger Verwendung des Temperaturwerts im Prozessmodell muss auch die Rückruffunktion wiederholt definiert werden, um jeweils den aktuellen Temperaturwert abzurufen. Außerdem ist die Implementierung durch die Einführung Camunda-spezifischer Attribute (`camunda:executionListener` in Abbildung 14) nicht BPMN-konform, wodurch es Portabilitätskonflikte bei der Verwendung in einer anderen Software geben kann. Zuletzt ist es in BPMN nicht erlaubt, einen Datenspeicher mit einem exklusiven Gatter zu verbinden. Dadurch kann die gewählte Implementierung in der grafischen Darstellung nicht dokumentiert werden.

```
<bpmn:exclusiveGateway>
    <bpmn:extensionElements>
        <camunda:executionListener event="end">
            <camunda:script scriptFormat="javascript">
                // Temperatur abrufen ...
            </camunda:script>
        </camunda:executionListener>
    </bpmn:extensionElements>
</bpmn:exclusiveGateway>
```

Abbildung 14: Ausschnitt der textuellen XML-Repräsentation des Prozesses zur Temperaturüberprüfung

2.5.5 *Die operationale Perspektive*

Die operationale Perspektive integriert die Prozessausführung in die IT-Systemlandschaft eines Unternehmens.

2.5.5.1 *Konzeptionelle Sichtweise*

Externe Anwendungen

Das Aufgabenspektrum in Unternehmen ist sehr heterogen und für verschiedene Aufgaben stehen verschiedene Anwendungen zur Verfügung. Beispielsweise werden jeweils spezialisierte Applikationen zur Mitarbeiterverwaltung, für die Verwaltung von Produkten oder zur Erledigung von Buchhaltungstätigkeiten eingesetzt. Für bestimmte Aufgaben können auch mehrere Anwendungen herangezogen werden, zum Beispiel Microsoft Word oder LaTeX zur Erzeugung eines Dokuments. Im Allgemeinen kann aus Sicht der operationalen Perspektive jeder ausführbare Quelltext als externe Anwendung betrachtet werden, im Speziellen sind ausführbare Abbilddateien, Funktionen von Programmbibliotheken oder Kommandozeilenskripte exemplarisch angegeben [78, S. 159]. Das Ziel der operationalen Perspektive ist unter anderem die automatische Auswahl geeigneter Anwendungen und deren automatisierter Aufruf beziehungsweise die automatisierte Ausführung des Quelltexts. Neben einer automatisierten Ausführung kann eine Anwendung andererseits auch interaktiv bedient werden (müssen). In diesem Fall muss die Umgebung die geeigneten Anwendungen zur Verfügung stellen, welche vom WFMS aufgerufen werden, und vom Benutzer dann im Rahmen der Abarbeitung der Aktivität bedient werden.

Operation und Anwendung

Im ursprünglichen Konzept erfolgt dabei eine Entkopplung zwischen den auszuführenden Operationen und den zur Verfügung stehenden implementierenden Anwendungen, wobei das WFMS geeignete Anwendungen für die auszuführenden Operationen automatisiert auswählt. Dafür ist analog zum Organisationsmodell eine strukturelle Beschreibung von Applikationen sowie deren Eigenschaften und bereitgestellten Operationen notwendig [78, S. 159]. Die Entscheidungsfindung für die Auswahl externer Anwendungen kann dabei auch von Informationen anderer Perspektiven beeinflusst werden, wie das folgende Beispiel zeigt.

Beispiel: Aktivität in der operationalen Perspektive. Ein Prozessmodell beinhaltet eine Aktivität Bericht anfertigen. In der organisationalen Perspektive wird festgelegt, dass ausschließlich Mitarbeiter mit der Rolle Sekretärin diese Aktivität ausführen dürfen und somit in der Arbeitsliste vorfinden. In der operationalen Perspektive wird weiterhin spezifiziert, dass diese Aktivität mithilfe einer externen Anwendung ausgeführt werden muss. Die Anwendung muss dabei vom Typ Textverarbeitungsprogramm sein. In einem operationalen Modell

können dann Microsoft Word und LaTeX als Instanzen dieses Typs definiert werden. Die Anwendungen werden weiterhin über Attribute beschrieben, nämlich dass die Bedienung von LaTeX Programmierkenntnisse erfordert, während die Verwendung von Microsoft Word einer Lizenz bedarf. Wenn nun im organisationalen Modell die Fertigkeiten und Lizenzen der Agenten mit berücksichtigt sind, kann das WFMS bei der Ausführung der Aktivität Bericht anfertigen entsprechend die geeignete Anwendung starten.

In dieser Arbeit wird die Ausdrucksmächtigkeit der operationalen Perspektive eingeschränkt betrachtet. Es soll weder das interaktive Bedienen von Anwendungen berücksichtigt werden, noch eine automatisierte Selektion von Anwendungen, welche über ein operationales Modell klassifiziert sind. Stattdessen liegt der Fokus auf den Funktionalitäten, welche über BPMN und Camunda zur Verfügung gestellt werden, nämlich die automatisierte Ausführung von Quelltext oder das Aufrufen von externen Computerprogrammen.

2.5.5.2 *Umsetzung in BPMN*

BPMN integriert mit *Service Tasks* und *Script Tasks* zwei Aktivitätstypen im Sinne der operationalen Perspektive.

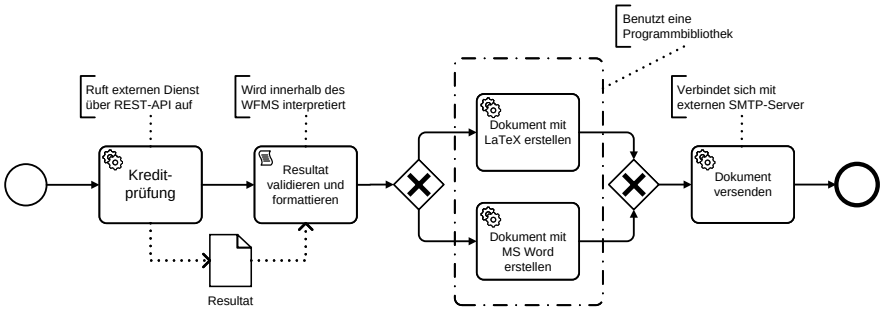

Abbildung 15: Operationale Perspektive: BPMN Service Tasks

SCRIPT TASK Der Aktivitätstyp *Script Task* (Resultat validieren und formatieren in Abbildung 15) beschreibt die WFMS-interne Interpretation eines Skripts [135, S. 162]. Das Skript wird direkt von der Ausführungsmaschine interpretiert und die Ausführung verlässt dabei nicht den Geltungsbereich des WFMS. Dementsprechend kann das Skript auch direkt im Prozessmodell abgespeichert werden, so wie in Abbildung 16 demonstriert.

```
<bpmn:scriptTask name="Resultat..." scriptFormat="JavaScript">
    ...
    <bpmn:script>
        function validate(result) {
            const requiredFields = ['name', 'score', 'approved'];
            for (let field of requiredFields) {
                if (!(field in result)) {
                    throw new Error('fehlendes Feld: ${field}');
        } } //...}
    </bpmn:script>
</bpmn:scriptTask>
```

Abbildung 16: Script Task Resultat validieren und formatieren mit dem auszuführenden Quelltext

SERVICE TASK Der Aktivitätstyp *Service Task* (unter anderem Kreditprüfung in Abbildung 15) beschreibt hingegen den Aufruf WFMS-externer Dienste. Service Tasks werden mit der zu verwendeten Technologie und einem Operationsparameter konfiguriert [135, S. 157]. Die Anbindung von externen Anwendungen und Quelltext ist eher auf der Ebene der Implementierung im Prozesslebenszyklus anzusiedeln, weswegen die auf die Modellierung fokussierende BPMN in dieser Hinsicht dementsprechend eingeschränkt ist [61]. Die Implementierungsverantwortung liegt nicht aufseiten des BPMN-Standards, sondern fällt in den Zuständigkeitsbereich der Programmierer und damit in den verwendeten Ausführungsmaschinen (Kapitel 2.5.5.3).[11] Das Prozessmodell in Abbildung 15 enthält vier Service Tasks, nämlich Kreditprüfung, Dokument mit LaTeX erstellen, Dokument mit MS Word erstellen und Dokument versenden, wobei keine Implementierungsdetails in der grafischen Darstellung erkennbar sind. Zur Dokumentation sind den Service Tasks lediglich Kommentare in Form von Annotationen angehängt, welche jedoch keine semantische Aussagekraft haben.

2.5.5.3 *Implementierung in Camunda*

Für die Implementierung von Service Tasks integriert das Camunda WFMS herstellerspezifische Strukturen in die XML-Repräsentation von BPMN-Diagrammen. In der XML-Repräsentation in Abbildung 17 zeigt sich das Camunda-spezifische `camunda:class`-Attribut, dessen Wert die Java-Klasse spezifiziert, dessen Methode bei der Ausführung aufgerufen wird. Die Java-Klasse enthält dann beispielsweise direkt die Programmlogik für die Kreditprüfung oder ruft über ein Protokoll, wie zum Beispiel HTTP, einen externen Dienst mit den benötigten Parametern auf und erwartet das Ergebnis in der entsprechenden HTTP-Antwort. Das Ergebnis kann dann in eine Prozessvaria-

11 https://www.trisotech.com/bpmn-decoded-data-flow/, besucht am 19.11.2023

ble gespeichert werden und im weiteren Verlauf der Prozessausführung weiterverarbeitet werden. Auf die Implementierungsdetails in Camunda wird auf dieser Stelle nicht näher eingegangen.

```
<bpmn:serviceTask id="1bw" name="Dokument versenden"
    camunda:class="de.ubt.ai4.creditcheck.SendDocument">
</bpmn:serviceTask>
<bpmn:textAnnotation id="0od">
    <bpmn:text>
        Verbindet sich mit externen SMTP-Server
    </bpmn:text>
</bpmn:textAnnotation>
<bpmn:association sourceRef="1bw" targetRef="0od" />
```

Abbildung 17: Service Task Dokument Senden mit implementierender Camunda-Klasse der BPMN-Annotation

3

ZWISCHENBETRIEBLICHES PROZESSMANAGEMENT

Kapitel 2 führt in die Grundlagen des Geschäftsprozessmanagements ein. Es wird darin der Prozessbegriff vorgestellt sowie Prozesse mit Aktivitäten und deren Lebenszyklen beschrieben. Dabei steht die konzeptionelle Prozessmodellierung und die Relevanz unterschiedlicher Perspektiven in Zusammenhang mit der Ausführung von Prozessmodellen in einem WFMS im Vordergrund. Die praktische Umsetzung der Konzepte ist am Beispiel von BPMN und Camunda demonstriert. Die Erläuterungen folgen dabei den Grundannahmen eines organisationsinternen Geltungsbereichs der Modelle und einer zentral verwalteten WFMS-Infrastruktur.

Das folgende Kapitel definiert darauf aufbauend zwischenbetriebliche Prozesse. Diese zeichnen sich dadurch aus, dass verschiedene organisationale Entitäten, zum Beispiel Abteilungen oder Unternehmen, an einem gemeinsamen Prozessziel beteiligt sind. Dabei treten zusätzliche Herausforderungen auf. Diese umfassen aus Modellierungssicht die Erstellung konsolidierter Prozess- und Organisationsmodelle unter Wahrung von Betriebsgeheimnissen oder des Schutzes personenbezogener Daten. In Hinblick auf die Implementierung und Ausführung hingegen ist eine Infrastruktur für die Verwaltung der Prozesskontrolldaten zu finden.

Kapitel 3.1 diskutiert die zusätzlichen Herausforderungen im Gegensatz zu den innerbetrieblichen Prozessen, insbesondere in Hinblick auf die Modellierung sowie auf Infrastrukturen zur Implementierung und der Ausführung. Grundsätzlich lassen sich für zwischenbetriebliche Prozesse die nachrichtenbasierten Modelle (Kapitel 3.2) von den prozessbasierten Modellen (Kapitel 3.3) unterscheiden. Für beide Modellierungsparadigmen werden dabei mit zentral verwalteten, verteilten und dezentralen Infrastrukturen jeweils drei verschiedene Ansätze für die Implementierung untersucht. Die möglichen Kombinationen werden in Kapitel 3.4 zusammenfassend diskutiert. Zuletzt werden in Kapitel 3.5 die Umsetzungsmöglichkeiten in Bezug auf die in verwandten Arbeiten vorgeschlagenen Arten von *Workflow-Interoperabilität* einsortiert und diese dabei in Hinblick auf die folgenden Kapitel um den neu eingeführten Typ *Decentralized Control* ergänzt.

3.1 EINFÜHRUNG IN ZWISCHENBETRIEBLICHE PROZESSE

Bei zwischenbetrieblichen Prozessen und deren Ausführung sind verschiedene Personen, Gruppen, Unternehmen oder Organisationen involviert, um an einem gemeinsamen Prozessziel zu arbeiten und dieses zu erreichen [187, S. 259]. Das zwischenbetriebliche Prozessmanagement tangiert somit nicht nur den Forschungsbereich des Geschäftsprozessmanagements, sondern inhäriert polyvalente Disziplinarität mit Intersektionen im sozial- und wirtschaftswissenschaftlichen Bereich. Im Folgenden wird in Kapitel 3.1.1 die Zusammenarbeit von organisationalen Entitäten kurz aus der Perspektive der Sozial- und Wirtschaftswissenschaften skizziert, bevor Kapitel 3.1.2 die zwischenbetrieblichen Prozesse im Kontext dieser Arbeit einführt.

3.1.1 *Sozialwissenschaftliche und betriebswirtschaftliche Sicht*

Kooperation versus Kollaboration

Wenn mehrere Teilnehmer an einem gemeinsamen Ziel zusammenarbeiten, unterscheidet man die Kooperation von der Kollaboration [147]. Dabei ist die Kooperation durch das Aufteilen von Arbeit unter den Teilnehmern definiert, wobei jeder Teilnehmer für einen bestimmten Teil der gesamten Arbeit zuständig ist. Bei einer Kollaboration steht hingegen das gemeinsame, koordinierte Lösen einer Herausforderung im Vordergrund. Andere Autoren verwenden die Begriffe ohne scharfe Trennung synonym, da auch bei kollaborativer Arbeit zeitweise eine Art der Trennung entstehen kann [38], sodass bestimmte Arbeitsschritte auch hier isoliert nur von bestimmten Teilnehmer erledigt werden. Aus einer wirtschaftswissenschaftlichen Sichtweise heraus kann die Kollaboration als eine Art der Kooperation verstanden werden, wie der folgende Abschnitt erläutert.

Formen der Kooperation

Es existieren verschiedene Formen, nach denen Unternehmen oder Organisationen in einer Kooperationsbeziehung stehen können. Mögliche Kooperationsformen sind unter anderem Allianzen, Kollaborationen, Netzwerke, Koalitionen, Konsortien, Partnerschaften oder Joint Ventures [158]. Die verschiedenen Ausprägungen der Geschäftsbeziehungen unterscheiden sich unter anderem in ihrer Organisation, ihrer Arbeitsweise und in den jeweils verfolgten Zielen. So erfolgt die Zusammenarbeit bei einem Joint Venture durch die Gründung eines rechtlich selbstständigen gemeinsamen Tochterunternehmens mit dem Ziel der Risikoverteilung und Nutzung von gegenseitigem Know-how [42]. Bei einer (Ad-hoc)-Koalition [51], welche als Projektgemeinschaft organisiert ist, arbeiten hingegen mehrere (rechtlich selbstständige) Unternehmen an einem gemeinsamen Projektziel. Der Begriff Allianz beschreibt auf einer abstrakten Ebene eine grundsätzliche Vereinbarung von Unternehmen zur Zusammenarbeit [52].

Anwendbarkeit zwischenbetrieblicher Prozessausführung

Die verschiedenen Organisationsmöglichkeiten und Ziele von Kooperationsformen deuten darauf hin, dass die in dieser Arbeit vorge-

stellte Form der zwischenbetrieblichen Prozessausführung nicht im allgemeinen Fall für alle Kooperationen gleichermaßen sinnvoll anwendbar ist. Somit ist ein möglicher Einsatz für jede Kooperation per se zu evaluieren. In weiterführender Forschungsarbeit kann überprüft werden, ob Aussagen über die Anwendbarkeit zumindest allgemeingültig für bestimmte Kooperationsformen getroffen werden können. Zum Beispiel hat die Kooperationsform der Kollaboration die Integration von unternehmensinternen und unternehmensübergreifenden Geschäftsprozessen zum Ziel [90], was durch die explizite Nennung der unternehmensübergreifenden Geschäftsprozesse auf eine erfolgreiche Anwendbarkeit hindeutet. Wenn andererseits bei einem Joint Venture eine rechtlich selbstständige Einheit neu gegründet wird, können die zur Erreichung der Prozessziele benötigten Serverinfrastrukturen, Ausführungssysteme und (menschliche) Ressourcen zentral vom gegründeten Unternehmen zur Verfügung gestellt werden. Damit können eher die Prinzipien der traditionellen, innerbetrieblichen Prozessausführung wie in Kapitel 2.3 beschrieben zum Einsatz kommen.

Das Ziel dieser Arbeit ist nicht, zu prüfen, ob und in welcher Form eine zwischenbetriebliche Prozessausführung bei den unterschiedlichen Kooperationsformen möglich ist und umgesetzt werden kann. Stattdessen definiert der folgende Abschnitt die Voraussetzungen, welche bei der in dieser Arbeit betrachteten zwischenbetrieblichen Kooperationen angenommen werden.

3.1.2 *Sichtweise des BPM-Forschungsbereichs*

Aus Sicht des Geschäftsprozessmanagements können zwischenbetriebliche Prozesse im Gegensatz zur innerbetrieblichen Prozessausführung nach Kapitel 2 auf unterschiedliche Weise konzipiert werden. Die zur Verfügung stehenden Optionen lassen sich anhand zweier Dimensionen strukturieren: Modellierung und implementierende Infrastrukturen. Aus Modellierungssicht können prozessbasierte Modelle von den nachrichtenbasierten Modellen unterschieden werden. Aus Implementierungssicht stehen mit zentral verwalteten, verteilten und dezentralen Infrastrukturen jeweils drei verschiedene Varianten zur Verfügung. Dadurch ergeben sich insgesamt sechs verschiedene Optionen für die Umsetzung zwischenbetrieblicher Prozesse, nämlich nachrichtenbasierte Modelle auf zentralen, verteilten und dezentralen Infrastrukturen beziehungsweise prozessbasierte Modelle auf zentralen, verteilten und dezentralen Infrastrukturen. Abbildung 18 bietet eine Übersicht über die Optionen und zeigt dabei die Strukturierung der folgenden Kapitel.

Im Anschluss werden zuerst die zwei Dimensionen vorgestellt, die Modellierung in Kapitel 3.1.2.1 und die implementierenden Infrastrukturen in Kapitel 3.1.2.2. Daraufhin beschreibt Kapitel 3.2 den

nachrichtenbasierten Ansatz, während Kapitel 3.3 den prozessbasierten Ansatz vorstellt. Darin werden jeweils die Modellierung (Kapitel 3.2.1 beziehungeweise Kapitel 3.3.1) und die möglichen Infrastrukturen (Kapitel 3.2.2 beziehungsweise Kapitel 3.3.2) diskutiert.

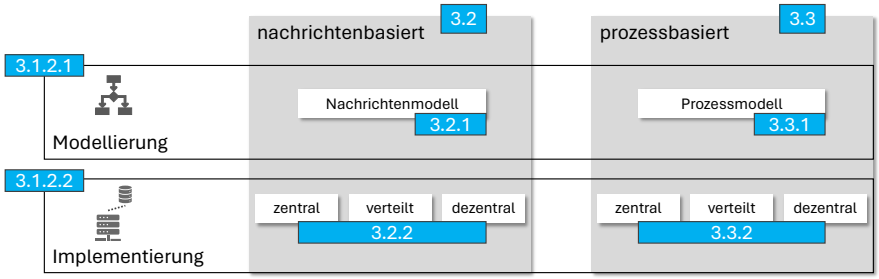

Abbildung 18: Strukturierung der folgenden Abschnitte

3.1.2.1 *Modellierungsaspekte zwischenbetrieblicher Prozesse*

Für die Implementierung und Umsetzung eines zwischenbetrieblichen Prozesses muss dieser ähnlich zur innerbetrieblichen Prozessausführung in eine modellhafte Form gebracht werden. Wie in Abbildung 19 dargestellt, können hierbei zwei Varianten unterschieden werden, welche sich im Grunde auf die unterschiedliche Definition von Kooperation und Kollaboration aus Kapitel 3.1.1 abbilden lassen.

nachrichtenbasierte Modelle

Ein nachrichtenbasiertes Modell stellt die erforderliche oder erwartete Interaktion aller Teilnehmer in Form von Nachrichten in den Vordergrund und fördert so das kooperative Arbeiten. Das bedeutet, jeder Teilnehmer arbeitet zunächst unabhängig an seiner zugewiesenen Teilaufgabe und nach deren Fertigstellung informiert er anhand des nachrichtenbasierten Modells den oder die nächsten Teilnehmer, welche dann wiederum unabhängig ihren Teil der Zusammenarbeit beitragen können. Die nachrichtenbasierten Ansätze forcieren damit die vollständige Auslagerung bestimmter Aufgaben oder Teilprozesse. Interne Details sind bei diesem Konzept für externe Teilnehmer weder sichtbar noch von Interesse.

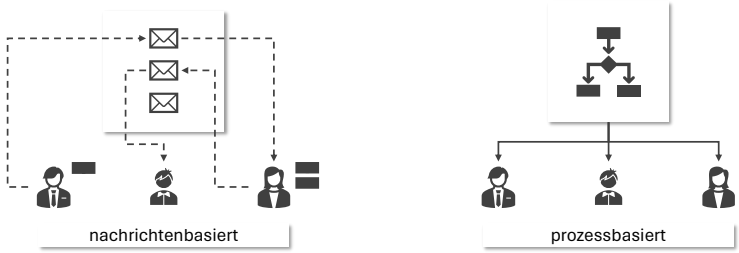

Abbildung 19: Modellierungsprinzipien zwischenbetrieblicher Prozesse

prozessbasierte Modelle

Der prozessbasierten Variante liegt ein konsolidiertes, gegebenenfalls multiperspektivisches Prozessmodell zugrunde, welches *alle* ein-

zelnen Arbeitsschritte aller Teilnehmer zur Erreichung des Kollaborationsziels enthält. Statt das zu befolgende Kommunikationsschema zu modellieren, beschreibt das Prozessmodell die erforderlichen Schritte für das Erreichen des Prozessziels.

3.1.2.2 *Infrastrukturen für zwischenbetriebliche Prozesse*

technische Voraussetzungen

Für die Umsetzung der betriebsinternen Prozessausführung existiert genau eine Instanz, welche aus technischer Sicht die IT-Infrastruktur kontrolliert, einen zentralen Server verwaltet und die darauf laufenden Anwendungen administriert. Die IT-Infrastruktur übernimmt sowohl die Bereitstellung eines WFMS als auch die dazugehörige Benutzerverwaltung sowie die Datenhaltung und -persistierung. Bei der zwischenbetrieblichen Prozessausführung ist die Situation gegensätzlich, da die beteiligten Organisationen jeweils eine eigene IT-Infrastruktur kontrollieren und administrieren können, sich jedoch keine gemeinsame IT-Infrastruktur teilen müssen, welche zur Bereitstellung der Funktionen verwendet werden kann.

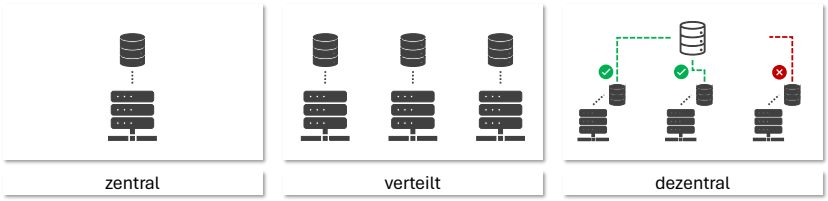

Abbildung 20: Mögliche Infrastrukturen der zwischenbetrieblichen Prozessausführung

Im zwischenbetrieblichen Umfeld können mit der zentralen Kontrolle, der verteilten Kontrolle und der dezentralen Kontrolle insgesamt drei verschiedene Varianten zur technischen Umsetzung identifiziert werden (Abbildung 20) [169]. Bei der zentralen Kontrolle wird entweder ein externer Drittanbieter oder einer der Teilnehmer für die Administration der IT-Infrastruktur beauftragt. Bei der verteilten Kontrolle verwenden die Teilnehmer jeweils unabhängig voneinander ihre lokale IT-Infrastruktur, welche dadurch allerdings nicht synchronisiert werden. Bei der dezentralen Kontrolle werden alle Daten auf den Infrastrukturen der Teilnehmer repliziert. Die lokalen Datenstände werden über fehlertolerante Synchronisationsprotokolle koordiniert, welche prinzipiell für einen global eindeutigen und akzeptierten Datenstand sorgen.

zentrale Kontrolle

verteilte Kontrolle

dezentrale Kontrolle

Die drei Varianten werden jeweils in Kapitel 3.2 und Kapitel 3.3 für die Beschreibung der möglichen Umsetzungen zwischenbetrieblicher Prozesse genauer diskutiert.

3.2 NACHRICHTENBASIERTE SYNCHRONISATION LOKALER PROZESSE

Konzept der nachrichtenbasierten Kollaboration

Anstatt der Ausführung eines konsolidierten, zwischenbetrieblichen Prozessmodells liegt der Fokus bei der nachrichtenbasierten Synchronisation lokaler Prozesse auf dem zu erfolgendem Nachrichtenaustausch, der für eine ordnungsgemäße Umsetzung der zwischenbetrieblichen Kollaborationen zu erfolgen hat. In der Umsetzung organisiert demnach jeder Teilnehmer der Kollaboration lokal die erforderlichen Schritte, welche jeweils zum Prozessziel beitragen und somit in ihrer Gesamtheit das Ziel des zwischenbetrieblichen Prozesses erreichen. Während der Durchführung kommunizieren die Teilnehmer dem Protokoll folgend über den Austausch von Nachrichten. Der Empfang einer Nachricht signalisiert einem wartenden Teilnehmer, dass die Kollaboration entsprechend vorangeschritten ist und er seinerseits die anstehenden Aufgaben abarbeiten kann bis er wiederum dem nächsten Teilnehmer eine Nachricht sendet. Der Empfang einer Nachricht kann dabei von jedem Teilnehmer vollkommen unabhängig und lokal unterschiedlich interpretiert und verarbeitet werden, wobei nicht zwingend ein WFMS zum Einsatz kommen muss. Obwohl auch andere Umsetzungsoptionen existieren, wird hier angenommen, dass die Implementierung über die jeweils intern ausgeführten Prozesse in den lokalen WFMSs der Teilnehmer erfolgt, etwa in der operationalen Perspektive über Service Tasks, wodurch die lokal ausgeführten Prozesse synchronisiert werden. Dieses Vorgehen wird in ähnlicher Weise auch von Weske et al. vorgeschlagen [187]. Im Folgenden beschreibt Kapitel 3.2.1 die Modellierungsartefakte, welche BPMN diesbezüglich zur Verfügung stellt, während Kapitel 3.2.2 die umsetzenden Infrastrukturen diskutiert.

3.2.1 *Modellierung der nachrichtenbasierten Kollaboration*

Statt der Verwendung eines Prozessmodells wird die zwischenbetriebliche Kollaboration hierbei über ein Nachrichtenaustauschprotokoll modelliert. Die Herausforderungen dabei sind einerseits die konforme Ableitung und Implementierung lokale Prozesse und andererseits die Sicherstellung der Kompatibilität und die Interpretierbarkeit von Nachrichten [187, S. 270]. Im Folgenden wird eine mögliche Vorgehensweise des Modellierungsvorgangs aus [187] sowie die daraus entstehenden Modellierungsartefakte beschrieben.

3.2.1.1 *Modellierungsvorgang*

Meilensteine und Teilnehmer

Um eine nachrichtenbasierte Kollaboration zu entwerfen und umzusetzen, wird üblicherweise ein *Top-down*-Ansatz verfolgt, nach dem zuerst das unternehmensübergreifende Ziel festgelegt wird, bevor einzelne Meilensteine definiert und die Teilnehmer der Kollabora-

tion inklusive deren notwendige Interaktionen identifiziert werden. Davon ausgehend werden für jeden Teilnehmer sogenannte Verhaltensschnittstellen (englisch: *behavioural interfaces*) extrahiert. Diese beschreiben die Kollaboration aus dessen Sicht und geben vor, welche Nachrichten während der Kollaboration gesendet und empfangen werden müssen. Die Verhaltensschnittstellen, also die Nachrichtenkommunikation, muss im letzten Schritt von den lokalen Prozessen implementiert werden.

Verhaltensschnittstellen

lokale Prozesse

Beispiel: Pizzakollaboration. Die oben genannten Schritte sollen nun anhand der Kollaboration einer Pizzabestellung exemplarisch erklärt werden.

In der Identifikationsphase werden der Kunde, die Pizzeria und der Lieferant als Teilnehmer ermittelt. Die Meilensteine umfassen die Bestellung der Pizza und die Lieferung beziehungsweise den Erhalt der Pizza. Für die Meilensteine werden dann die notwendigen Interaktionen extrahiert, zum Beispiel ist bei der Pizzabestellung eine Interaktion zwischen dem Kunden und der Pizzeria erforderlich. Aus diesen Interaktionen wird das Nachrichtenaustauschmodell abgeleitet, welches beispielsweise über eine BPMN-Choreografie abgebildet werden kann (Abbildung 21). Damit ist die Kollaboration bis auf die genaue Spezifikation der Nachrichtenformate vollständig definiert.

Die darauffolgende Implementierung der lokalen Prozesse erfolgt auf Basis der Verhaltensschnittstellen, welche die jeweils notwendige Interaktion eines bestimmten Teilnehmers mit allen anderen Teilnehmern repräsentiert (Abbildung 22). Die Verhaltensschnittstelle der Pizzeria beinhaltet das Erhalten der Bestelldetails sowie das Übergeben der fertigen Pizza an den Kunden oder gegebenenfalls an den Lieferanten. Basierend auf dieser Schnittstelle kann die Pizzeria einen privaten Prozess ableiten, der mindestens die Aktivitäten (oder Ereignisse) für den Erhalt der Bestellung sowie für die Aushändigung an den Kunden beziehungsweise an den Lieferanten enthält. Des Weiteren können noch weitere, interne Aktivitäten wie das Zubereiten der Pizza im privaten Prozess modelliert und implementiert werden.

3.2.1.2 *Modellierungsartefakte*

Dieses Kapitel stellt die essenziellen Konstrukte vor, welche BPMN für die Modellierung von zwischenbetrieblichen Kollaborationen zur Verfügung stellt, wofür das Beispiel der Pizzakollaboration fortgesetzt wird. Die Modellierungsartefakte lassen sich als *Interaction Model* (deutsch: Interaktionsmodell) und *Interconnected interface behaviour model* (deutsch: Schnittstellenverhaltensmodell) differenzieren [32].

INTERAKTIONSMODELL: BPMN CHOREOGRAFIE Ein Interaktionsmodell beschreibt die Reihenfolge, in welcher die Nachrichten ausge-

tauscht werden. Dabei wird sowohl der Nachrichtenaustausch per se benannt und modelliert als auch der jeweilige Sender und Empfänger der Nachricht. Das Modell beschreibt die Sichtweise eines außenstehenden Beobachters.

BPMN Choreografie

In Abbildung 21 ist eine BPMN-Choreografie abgebildet, welche den Nachrichtenaustausch zwischen den drei Teilnehmern der Pizzakollaboration modelliert.

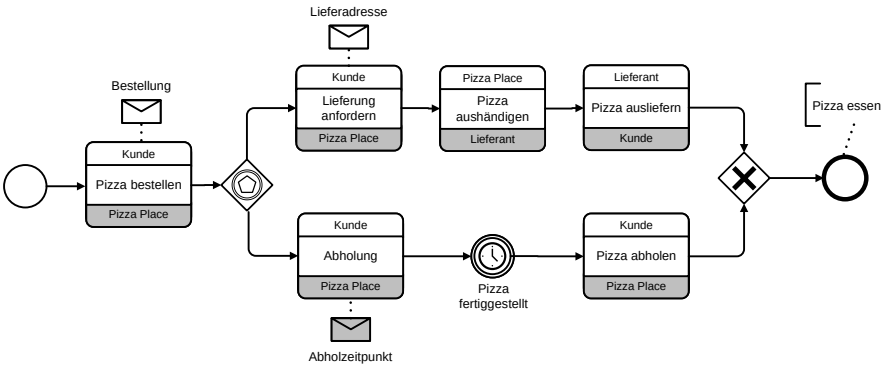

Abbildung 21: Pizzakollaboration als BPMN-Choreografie

Eine Choreografie setzt sich aus mehreren Choreografieaktivitäten zusammen, welche jeweils eine Interaktion zwischen zwei oder mehreren Teilnehmern beschreibt und neben den Namen der Interaktion auch den Sender (englisch: *Initiator*) und den jeweiligen Empfänger (englisch: *Respondent*) modelliert. Die Choreografie in Abbildung 21 beginnt mit dem Nachrichtenaustausch Pizza bestellen, welche vom Teilnehmer Kunde initiiert wird und an den Teilnehmer Pizza Place gerichtet ist. Darüber hinaus wird noch die übermittelte Nachricht (Bestellung) benannt, jedoch nicht näher hinsichtlich der erwarteten Struktur spezifiziert.

BPMN Kollaborationsdiagramm

SCHNITTSTELLENVERHALTENSMODELL: BPMN KOLLABORATIONSDIAGRAMM Die Choreografie wird nach dem BPMN-Standard nicht direkt, etwa in einem WFMS, interpretiert, sondern dient als Dokumentationsartefakt, um weitere Diagramme für die Implementierung lokaler Prozesse abzuleiten. Eine entscheidende Rolle spielt dabei das BPMN-Kollaborationsdiagramm, welches die Interaktionen innerhalb der Kollaboration mit den Verhaltensschnittstellen der Teilnehmer in Verbindung bringt und somit die Reihenfolge des Nachrichtenaustausches jeweils aus der Sicht eines bestimmten Teilnehmers modelliert.

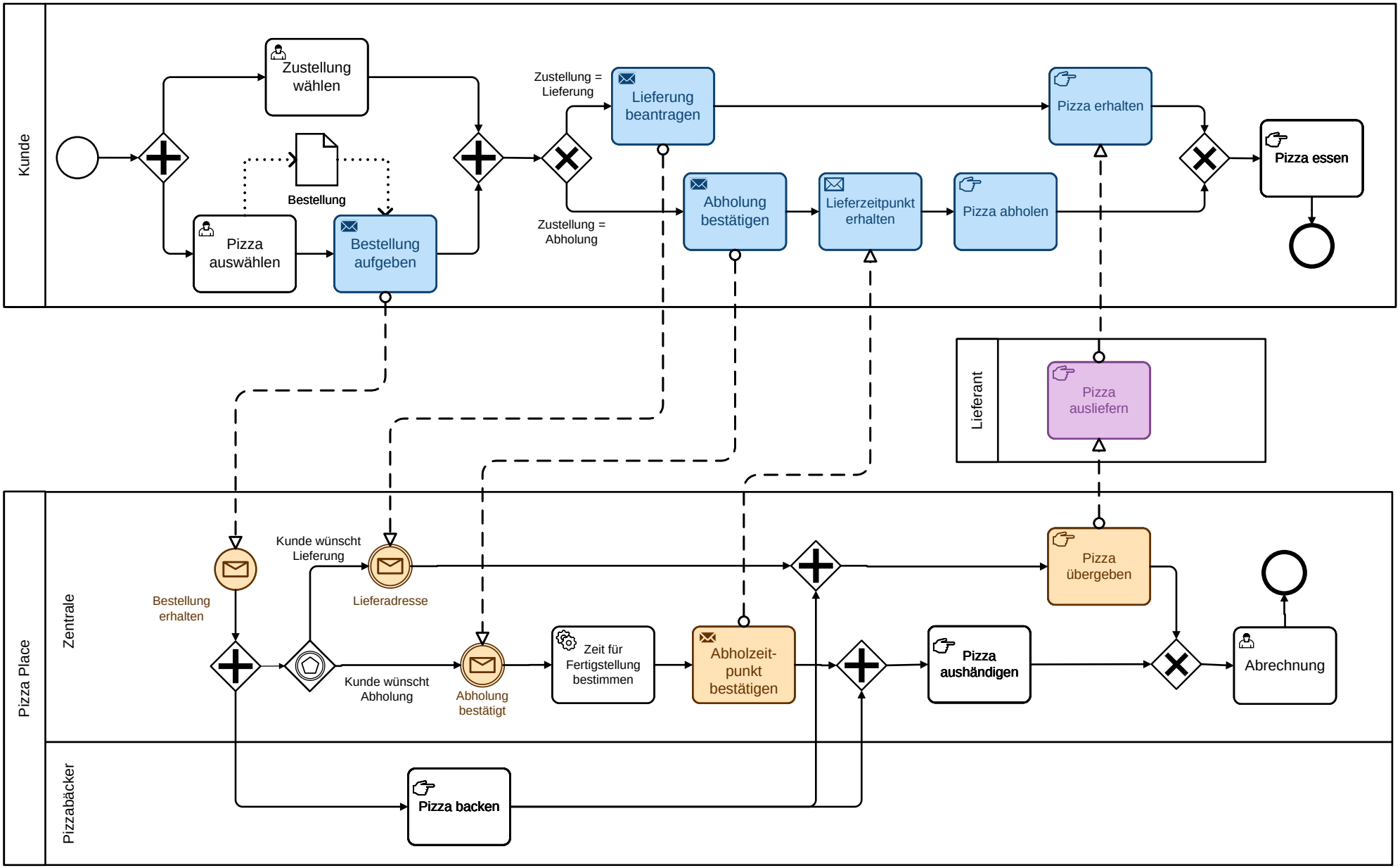

Abbildung 22: Pizzakollaboration als BPMN-Kollaborationsdiagramm

Abbildung 22 zeigt das Kollaborationsdiagramm der Pizzakollaboration. Darin ist jeder Teilnehmer, Kunde, Pizza Place und Lieferant, in einem separaten BPMN-Pool dargestellt, welcher den jeweiligen internen Prozess enthält.[1] Die zur Verhaltensschnittstelle gehörenden farbig dargestellten BPMN-Elemente markieren dabei die öffentlich sichtbaren Aktivitäten, über welche der Nachrichtenaustausch abgewickelt wird. Zum Beispiel substituiert die erste Choreografieaktivität (Pizza bestellen in Abbildung 21) den durch den dunklen Briefumschlag dekorierten Message Send Task Bestellung aufgeben und das Nachrichten-Startereignis[2] Bestellung erhalten im Pool des Pizza Place. Beim Pizza Place teilt das parallele Gatter nach der Instanziierung den Prozessfluss und startet einerseits die manuelle Aktivität Pizza backen.[3] Andererseits liegt ein weiteres Token an dem Ereignisbasierten Gatter, welches auf das Eintreffen einer Nachricht wartet. Deren Typ entsprechend (Lieferadresse oder Abholung bestätigt) wird der weitere Prozessfluss dann gesteuert.

Damit keine irrelevanten oder sensiblen Informationen offengelegt werden, enthalten Kollaborationsdiagramme üblicherweise lediglich die Verhaltensschnittstellen der anderen Teilnehmer und stellen deren Pools als Black Box dar.

3.2.2 *Implementierung der nachrichtenbasierten Kollaboration*

IMPLEMENTIERUNG MIT ZENTRALER KONTROLLE Aus technischer Sicht ist es möglich, die BPMN-Choreografie direkt als Artefakt für die Umsetzung einer Kollaboration zu nutzen, allerdings ist dies bei der Spezifikation der Sprache nicht berücksichtigt. Die Choreografie ist für Anwendungsfälle entworfen, bei der keine zentrale verantwortliche Kontrollinstanz verfügbar ist [135, S. 23]. Angesichts dessen ist eine Implementierung oder direkte Umsetzung der Choreografie nicht vorgesehen.

Choreografie-Ausführung über Nachrichtenvermittler

Nichtsdestotrotz kann eine zentral verwaltete Instanz als eine Art Nachrichtenvermittler (englisch: *message broker*) fungieren. Damit würden alle Nachrichten, die dem Choreografiemodell nach ausgetauscht werden müssen, nicht direkt an die jeweiligen Empfänger gesendet werden, sondern zuerst an den zentralen Vermittler übergeben werden. Der Vermittler implementiert die Choreografie und leitet während der Ausführung die Nachrichten dem Choreografiemodell entsprechend an die endgültigen Empfänger weiter. Dabei werden die Nachrichten einer bestimmten *Instanz* der Kollaboration zugeordnet.

1 Der Pizza Place-Pool ist zur internen organisationalen Zuweisung der Aktivitäten weiterhin in zwei BPMN-Lanes, Zentrale und Pizzabäcker, geteilt.

2 Mithilfe eines Nachrichten-Startereignisses wird ein Prozess nicht manuell, sondern über eine eingehende Nachricht instanziiert.

3 Manuelle Aktivitäten, gekennzeichnet durch das Hand-Symbol, werden außerhalb des WFMS-Kontextes ausgeführt und werden von Camunda übersprungen, indem das ankommende Token direkt weitergeleitet wird.

Dies hat den Vorteil, dass über den Vermittler theoretisch der aktuelle Fortschritt der Choreografie abgefragt werden kann. Allerdings bleibt zu diskutieren, ob die zentrale Koordinierungsinstanz statt das nachrichtenbasierte Modell zu implementieren, nicht auch ein möglicherweise über die Perspektiven ausdrucksmächtigeres Prozessmodell ausführen könnte. Das Fehlen eines Koordinators ist die ursprüngliche Motivation hinter der Entstehung der Choreografie.

Verteilte Ausführung ohne synchronisiertem Datenstand

IMPLEMENTIERUNG MIT VERTEILTER KONTROLLE Die technische Umsetzung der Kollaboration mit einer verteilten Infrastruktur erfolgt durch die Implementierung der privaten Prozesse in den internen WFMSs der jeweiligen Teilnehmer und den Nachrichtenaustausch zwischen diesen.

Während der Durchführung der Kollaboration wird ein WFMS eines Teilnehmers dem Choreografiemodell folgend benachrichtigt. Dies führt entweder zur Instanziierung eines Prozessmodells, welches anschließend ausgeführt wird, oder zur Wiederaufnahme eines bereits instanziierten Prozesses, welcher auf eine eingehende Nachricht wartet. In beiden Fällen werden die anstehenden Aktivitäten lokal entsprechend dem Modell wie in Kapitel 2 beschrieben ausgeführt. Sobald alle zu erledigenden Teilaufgaben der zwischenbetrieblichen Kollaboration im Rahmen dieses lokalen Prozesses ausgeführt sind, benachrichtigt, beispielsweise über einen Message Send Task, die ausführende Organisation den entsprechenden Kollaborationspartner, der für die nächsten Teilschritte der zwischenbetrieblichen Kollaboration verantwortlich ist.

Enforceable versus Non-Enforceable

Sowohl bei der Modellierung der Choreografie als auch bei der Ableitung der jeweiligen privaten Prozesse sind bestimmte Herausforderungen zu beachten. Ist der Sender eines Nachrichtenaustausches nicht im vorherigen Nachrichtenaustauschschritt beteiligt, erfolgt keine automatisierte Benachrichtigung über die erfolgreiche Durchführung des vorangegangenen Nachrichtenaustausches. Dieses Konzept sieht zuerst weder eine globale Überwachung des Fortschritts vor, um den betroffenen Sender aktiv zu informieren, noch erfolgt eine Protokollierung über ein konsolidiertes Ereignisprotokoll, über welches die Teilnehmer den aktuellen Fortschritt überprüfen können. Die Literatur spricht in diesem Fall von einer nicht durchführbarern Choreografie (englisch: *non-enforceable Choreography*) [187]. Als Lösung wird die Anpassung des Nachrichtenaustauschmodells vorgeschlagen oder der Einsatz eines Nachrichtenvermittlers, welcher die Choreografie interpretiert und den nächsten Sender aktiv informiert. Dies kann zum Beispiel auf dezentralen Infrastrukturen über Blockchain-Protokolle umgesetzt werden [96]. Außerdem besteht bei einer willkürlichen Definition des Protokolls und der abgeleiteten lokalen Prozesse die Gefahr von Deadlocks. Als Lösung wird das auf Petri-Netzen basierende *Public-to-private*-Verfahren vorgeschlagen [2], welches über

Deadlocks

Konsistenzkriterien dafür sorgt, dass alle Teilnehmer ihre lokalen Prozesse in einer Art und Weise ableiten können, sodass diese in ihrer Gesamtheit eine stabile Abarbeitung der zuvor definierten globalen Choreografie ermöglichen [187, S. 279].

Blockchain als Nachrichtenvermittler

IMPLEMENTIERUNG MIT DEZENTRALER KONTROLLE Der obige Abschnitt zur Implementierung mit zentraler Kontrolle hat die Verwendung von nachrichtenbasierten Modellen infrage gestellt und den Einsatz von ausdrucksmächtigeren Prozessmodellen favorisiert. Analog ist auch bei der dezentralen Kontrolle ein logisch gemeinsamer Datenstand verfügbar, was den Einsatz eines Prozessmodells und dessen Ausführung nach Kapitel 2 ermöglicht. Dennoch werden auf Basis der dezentralen Blockchain-Technologie Ansätze vorgestellt, welche ein BPMN-Choreografiemodell zur direkten Ausführung über einen Nachrichtenvermittler einsetzen [96, 111]. In den Veröffentlichungen steht dabei weniger die Integration in die zwischenbetriebliche Prozessausführung im Fokus, sondern eher die technische Umsetzung mittels eines Blockchain-Protokolls. Die Autoren argumentieren für den Einsatz einer BPMN-Choreografie aufgrund des hohen Abstraktionsniveaus.

3.3 PROZESSBASIERTE AUSFÜHRUNG

Konzept der prozessbasierten Kollaboration

Anstelle der BPMN-Prozessdiagramme werden für zwischenbetriebliche Kollaboration nachrichtenbasierte Modelle wie etwa BPMN-Choreografien und daraus abgeleitete Kollaborationsdiagramme vorgeschlagen, welche auf Basis einer verteilt kontrollierten Infrastruktur umgesetzt werden können. Es ist allerdings auch für eine zwischenbetriebliche Kollaboration möglich, eine Prozessausführung mit einem Prozessmodell ähnlich zur innerbetrieblichen Umsetzung zu implementieren. Diesbezüglich wird im Folgenden ebenfalls für die unterschiedlichen Infrastrukturtypen erörtert, inwieweit diese sich für die Interpretation eines zwischenbetrieblichen Prozessmodells beziehungsweise für die Bereitstellung eines WFMS eignen. Dafür werden wiederum zuerst die organisatorischen Herausforderungen der Modellierung beschrieben, worauf dann die Auswahl einer geeigneten Strategie zur Implementierung diskutiert wird.

3.3.1 *Modellierung eines zwischenbetrieblichen Prozesses*

Grundlage für die prozessbasierte Ausführung ist die Erstellung eines Prozessmodells, welches die zwischenbetriebliche Kollaboration abbildet. Dazu zählen die Identifikation aller Aktivitäten, die Festlegung deren zeitlicher Abarbeitungsreihenfolge und die optionale Integration verschiedener Prozessperspektiven, so wie in Kapitel 2.5 vorgestellt.

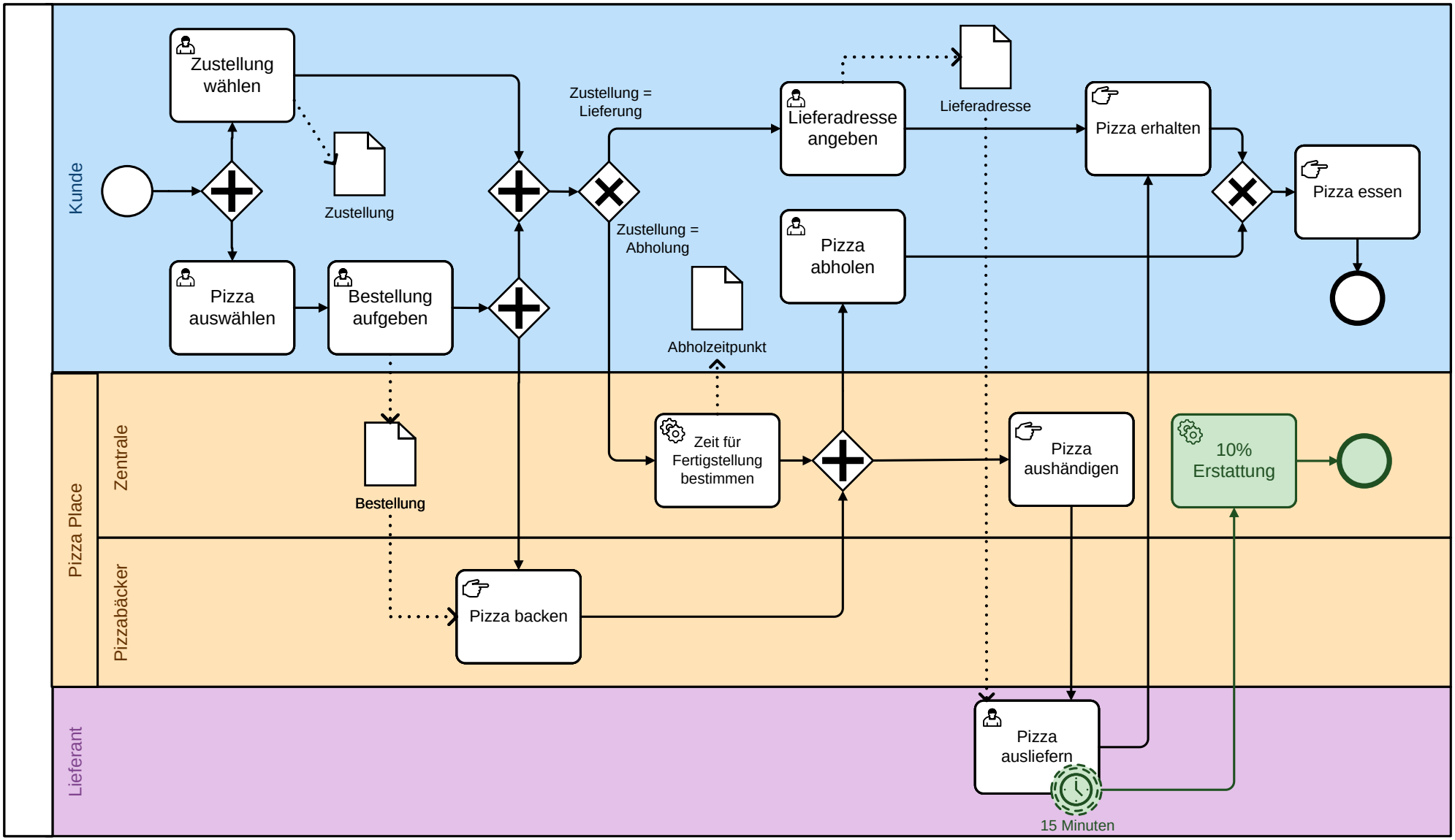

Abbildung 23: Pizzakollaboration als BPMN-Prozessdiagramm

Pizzakollaboration als Prozessdiagramm

Abbildung 23 zeigt eine Möglichkeit, wie die Pizzakollaboration als BPMN-Prozessmodell umgesetzt werden kann. Durch die Modellierung als Prozess und der Integration von verschiedenen Prozessperspektiven sind dabei noch weitere Verbesserungen und Unterstützungen möglich. Die folgenden Beispiele demonstrieren exemplarisch die Möglichkeiten der prozessbasierten Kollaboration gegenüber der Verwendung eines nachrichtenbasierten Modells (Kapitel 3.2.1). Zum einen können die Aktivitäten als User Task modelliert werden, sodass anhand des globalen Organisationsmodells und unter Berücksichtigung des Aktivitätslebenszyklus eine kollaborationsweite Koordinierung der Arbeitslast über personalisierte Arbeitslisten ermöglicht wird. Zum anderen demonstrieren die grün dargestellten BPMN-Elemente im Prozess in Abbildung 23 die Integration der zeitlichen und operationalen Perspektive. Der Aktivität Pizza aushändigen wird dabei ein sogenanntes *Non-Interrupting Timer Boundary Event* (deutsch: Nicht unterbrechendes Zeitgrenzereignis) mit folgender Semantik hinzugefügt: Es wird die Zeit ab dem Zeitpunkt des Ankommens des Tokens bei der Aktivität gemessen. Sobald die konfigurierte Dauer von 15 Minuten erreicht ist (zeitliche Perspektive), erzeugt das Ereignis automatisch ein weiteres Token, welches über die Aktivität 10% Erstattung dem Kunden aufgrund einer langen Lieferzeit einen Rabatt gewährt (operationale Perspektive). Ein nicht unterbrechendes Ereignis bricht dabei die angeheftete Aktivität nicht ab, sodass der Lebenszyklusstatus der Aktivität Pizza ausliefern nicht beeinflusst wird. Zuletzt können in einem weiterführenden Organisationsmodell darüber hinaus noch verschiedene Lieferanten organisiert werden, welche dann je nach Verfügbarkeit oder sogar entsprechend ihrer geografischer Position einer Bestellung zugewiesen werden können. Durch das Fehlen der Prozesskontrolldaten und der eingeschränkten Modellierungsoptionen können derartige Anforderungen bei nachrichtenbasierten Modellen nicht direkt umgesetzt werden.

Zusammengefasst deuten die Beispiele die Möglichkeiten an, welche durch die Modellierung der Pizzakollaboration als Prozessmodell entstehen.

3.3.2 *Implementierung einer prozessbasierten Kollaboration*

Nachdem das zwischenbetriebliche Prozessmodell erstellt ist, muss eine für die Kollaboration geeignete Infrastruktur für dessen Ausführung gefunden werden, welche entweder ein WFMS zur Verfügung stellt oder zumindest die Aufgaben eines WFMS übernimmt und dessen Funktionalitäten bereitstellt. Im Folgenden werden mit der zentral verwalteten, der verteilten und der dezentral verwalteten Ausführung die drei Optionen zur Umsetzung wieder kurz diskutiert.

ZENTRAL VERWALTETE AUSFÜHRUNG Bei der zentral verwalteten Ausführung wird das zwischenbetriebliche Prozessmodell in einem WFMS implementiert, welches auf einem zentral verwalteten Server bereitgestellt wird. Aus technischer Sicht ist diese Variante ähnlich zur innerbetrieblichen Prozessausführung, allerdings stellt sich die Frage nach der Verantwortlichkeit bezüglich der Bereitstellung und Administration des zentral verwalteten Servers. Mögliche Optionen bieten sich einerseits über die Auslagerung und der Einbindung eines externen Anbieters. Andererseits kann auch ein Teilnehmer der Kollaboration mit der Verwaltung der Infrastruktur beauftragt werden. Die beiden Varianten sind je nach Anwendungsfall hinsichtlich Sicherheit, Kosten und Aufwand zu evaluieren. Außerdem muss analog zur Integration der organisationalen Perspektive in der Modellierungsphase, in der Implementierung die Herausforderung der Benutzerverwaltung gelöst werden. Demnach müssen alle potenziellen Akteure der Prozessausführung über einen externen Verzeichnisdienst oder über eine WFMS-interne Benutzerverwaltung authentifiziert werden können.

extern versus intern

Authentifizierung

VERTEILTE AUSFÜHRUNG Eine verteilte Ausführung mit verteilten, nicht synchronisierten Datenständen widerspricht dem Grundgedanken eines konsolidierten Prozessmodells und dessen koordinierter Ausführung. Die verteilte Ausführung könnte dennoch implementiert werden, indem beispielsweise eine Prozessinstanz inklusive aller benötigter Daten von einem Teilnehmer zum nächsten weitergereicht wird und dort jeweils in ein lokales WFMS geladen und darüber weiter ausgeführt wird. Allerdings ist dabei eine dynamische Verteilung von Aktivitäten oder eine globale Überwachung schwer umzusetzen, sodass diese Variante hier nicht weiter betrachtet werden soll.

verteilte, nicht synchronisierte Datenstände

DEZENTRAL VERWALTETE AUSFÜHRUNG Eine alternative Strategie zur Implementierung und Ausführung bieten dezentral verwaltete Infrastrukturen. Dabei wird das zwischenbetriebliche Prozessmodell ohne eine Beteiligung einer zentralisierten Koordinierungsinstanz ausgeführt.

dezentrale Prozesskontrolldatenverwaltung

Die grundlegende Idee hinter der dezentral verwalteten Ausführung ist eine replizierte, synchronisierte Verwaltung der Prozesskontrolldaten in einem dezentralen Netzwerk unter Einsatz algorithmischer Konsensfindung. Dabei wird das zwischenbetriebliche Prozessmodell jeweils in den lokalen WFMSs der Teilnehmer interpretiert. Für die Umsetzung müssen alle prozessbezogenen Aktionen, welche den Prozessstatus aktualisieren, organisationsübergreifend an alle Teilnehmer kommuniziert werden. Die algorithmische Konsensfindung löst dabei die Herausforderungen bezüglich verteilter Systeme, wodurch sich alle Teilnehmer auf eine global eindeutige und akzeptierte Reihenfolge aller Aktionen einigen. Anschließend können diese

in den lokalen WFMSs interpretiert werden, wodurch jeder Teilnehmer zu jedem Zeitpunkt denselben Prozessstatus ableiten kann.

Die dezentral verwaltete Ausführung ist in dieser Arbeit von zentraler Bedeutung. In Kapitel 4 wird die Blockchain-Technologie als Vertreter einer dezentral organisierten Infrastruktur eingeführt. Kapitel 5 beschreibt darauf aufbauend eine Möglichkeit zur dezentralen Prozessausführung auf Basis der Ethereum-Blockchain, während Kapitel 6 ein allgemeines Rahmenwerk zur Integration beliebiger Synchronisierungsalgorithmen und WFMSs bereitstellt.

3.4 DISKUSSION

In Kapitel 3.2 und Kapitel 3.3 werden verschiedene Umsetzungsmöglichkeiten für zwischenbetriebliche Prozesse vorgestellt. Diese werden im Folgenden anhand bestimmter Kriterien innerhalb den beschreibenden Dimensionen Modellierung (Kapitel 3.4.1) beziehungsweise Implementierung (Kapitel 3.4.2) diskutiert.

3.4.1 *Modellierung*

Dieses Kapitel diskutiert die beiden Modellierungsansätze (prozessbasiert und nachrichtenbasiert) anhand der Kriterien Prozessperspektiven, Koordination, Betriebsgeheimnisse, Datenschutz, Flexibilität sowie Kompatibilität.

PROZESSPERSPEKTIVEN Beim prozessbasierten Ansatz wird die zwischenbetriebliche Kollaboration basierend auf einem Prozessmodell definiert, so wie in Kapitel 2 vorgestellt. Dadurch lassen sich im Gegensatz zum nachrichtenbasierten Modell verschiedene Aspekte in den Prozessperspektiven abbilden, wodurch komplexere Sachverhalte modelliert werden können. Dies ist bei der automatisierten Gewährung eines Rabatts bei verzögerter Lieferzeit gezeigt. Derartige Konstrukte, wenn etwa die Dauer einzelner Aktivitäten ausschlaggebend ist, können mittels Choreografien nur schwer umgesetzt werden.

Hinsichtlich der organisationalen Perspektive stellt sich jedoch die Frage, inwieweit das zur Umsetzung benötigte Organisationsmodell erstellt werden kann. Denn das erfordert die Definition aller möglichen Prozessteilnehmern über alle Unternehmen hinweg mit optionaler Zuweisung von Rollen oder sonstigen Attributen.

KOORDINATION Ein Prozessdiagramm definiert im Gegensatz zum Nachrichtenaustauschmodell alle Aktivitäten explizit und ermöglicht so deren kollaborationsübergreifende Koordination. Beim Einsatz eines Prozessmodells kann zum Beispiel eine Aktivität Pizza ausliefern von unterschiedlichen organisationalen Entitäten (Lieferanten) beansprucht werden. Derartige Funktionalitäten sind bei nachrichtenba-

sierten Modellen nicht möglich, sodass keine Arbeitslisten und damit keine kollaborationsübergreifende Koordination umgesetzt werden können.

BETRIEBSGEHEIMNISSE Während das Nachrichtenmodell von internen Abläufen der kollaborierenden Unternehmen abstrahiert, besteht durch die explizite Auflistung der Arbeitsschritte im Prozessmodell die Gefahr der Offenlegung von Betriebsgeheimnissen etwa über Produktionsverfahren. Es ist somit von den jeweiligen Teilnehmern stets auf eine ausreichend hohe Abstraktionsschicht zu achten, ohne dabei die Prozessausführung durch eine ungenügende Bereitstellung von Informationen zu beeinträchtigen.

DATENSCHUTZ Analog zur Offenlegung von Interna im Prozessmodell ist in ähnlicher Weise auch das Organisationsmodell betroffen, da durch die Auflistung der potenziellen Prozessakteure personenbezogene Daten offengelegt werden können. Die unternehmensübergreifende Veröffentlichung von Informationen zu menschlichen Ressourcen kann potenziell durch Pseudonymisierungsschritte vermieden werden, sodass zum Beispiel datenschutzrechtliche Richtlinien eingehalten werden können.

FLEXIBILITÄT Bei zwischenbetrieblichen Kollaborationen sind unter anderem die Autonomie und Unabhängigkeit der Teilnehmer als essenzielle Anforderungen definiert [106]. Diese kann durch die strikte Form eines Prozessmodells eingeschränkt werden: Sowohl Änderungen an internen Abläufen als auch personelle Umstrukturierungen in Unternehmen können Auswirkungen auf das globale Prozess- beziehungsweise Organisationsmodell haben. Durch eine weniger detaillierte Modellierung der internen Arbeitsschritte kann jedoch trotzdem die Flexibilität erreicht werden, interne Produktionsabläufe beliebig anzupassen. Über die Bildung von Subprozessen haben die Teilnehmer dann hierbei die Möglichkeit, den unterspezifizierten Prozessschritt im globalen Modell in einem privaten, lokal implementierten Prozess zu verfeinern.

Das nachrichtenbasierte Modell schränkt die Flexibilität der Prozessteilnehmer durch die hohe Abstraktionsschicht prinzipiell nicht ein. Die lokale Abarbeitung der relevanten Teilschritte kann jederzeit nach Bedarf unabhängig vom Choreografiemodell modell- und implementierungsseitig angepasst werden. Es muss lediglich sichergestellt werden, dass alle Nachrichten entsprechend der Choreografie empfangen und gesendet werden können.

KOMPATIBILITÄT Hinsichtlich der Verwendung von BPMN ist schließlich noch anzumerken, dass der Standard kein Diagram für die Erstellung eines zwischenbetrieblichen Prozessmodells zur Verfügung

stellt, da darin zwischenbetriebliche Kollaborationen mithilfe von nachrichtenbasierten Modellen beschrieben werden. Erste Versuche zeigen, dass die für interne Unternehmensabläufe bestimmten BPMN-Prozessdiagramme verwendet werden können (Kapitel 8), jedoch sind etwa Kompatibilitätsprobleme dadurch nicht auszuschließen.

FAZIT Während das nachrichtenbasierte Modell durch eine hohe Abstraktionsschicht und die dadurch einhergehende Flexibilität der Prozessteilnehmer überzeugt, ermöglicht das prozessbasierte Modell die Integration verschiedener Prozessperspektiven und dadurch die kollaborationsweite Koordination von Arbeitsschritten. Mögliche Einschränkungen durch die explizite Form des Prozessmodells lassen sich durch eine Abstraktion von internen Arbeitsabläufen umgehen. Auch die organisationale Perspektive kann in einer sehr abstrakten Detailstufe modelliert werden, indem die Prozessschritte lediglich den zuständigen teilnehmenden Organisationen entsprechend der verantwortlichen Sender und Empfänger im nachrichtenbasierten Modell zugewiesen werden und auf eine weiterführende Spezialisierung der Mitarbeitenden verzichtet wird. Ähnlich beim Prozessmodell gilt jedoch auch hier: Je spezifischer das Organisationsmodell definiert werden kann, desto wertschöpfender erfolgt möglicherweise die Koordination über die Prozessausführung.

3.4.2 *Infrastruktur*

Dieses Kapitel diskutiert die drei zur Verfügung stehenden Infrastrukturen für die zwischenbetriebliche Prozessausführung, bezeichnet durch die zentrale, verteilte und dezentrale Kontrolle, anhand der Kriterien Überwachung, Administration und Sicherheit.

ÜBERWACHUNG Die Überwachung des Gesamtfortschritts der zwischenbetrieblichen Kollaboration ist sowohl bei einem zentral verwalteten als auch in einem dezentralen Netzwerk möglich. Dies kann unabhängig vom Modellierungsansatz erfolgen, indem zum Beispiel das Ereignisprotokoll einer Prozessausführung abgerufen oder der bereits erfolgte Nachrichtenaustausch betrachtet wird. Bei einer zentral verwalteten Infrastruktur ist die Datenhaltung physisch zentralisiert, während beim Einsatz einer dezentralen Infrastruktur die Daten physisch repliziert vorliegen. Sie werden dabei durch geeignete Algorithmen synchronisiert, womit eine virtuell oder logisch einheitliche Datenbasis bereitgestellt wird. In einem verteilten Netzwerk hingegen, worin jeder Teilnehmer lediglich seinen eigenen Beitrag zum Gesamtprozess verwaltet, kann hingegen kein globaler Fortschritt abgeleitet werden.

ADMINISTRATION Die Frage nach der Administration stellt sich bei der Variante der zentralen Infrastruktur. Es bestehen dabei zwei Möglichkeiten. Entweder wird die zentrale Infrastruktur von einem der Kollaborationsteilnehmer zur Verfügung gestellt oder die Verwaltung wird an einem externen Drittanbieter ausgelagert.

Bei der verteilten Kontrolle verwaltet jede Organisation seine eigene Infrastruktur. Hierbei erfolgt die Administration jeweils intern in den Organisationen, welche für die Zusammenarbeit lediglich Schnittstellen oder Kommunikationsprotokolle festlegen müssen. Ähnlich erfolgt bei der dezentralen Kontrolle die Administration des Netzwerks und der Daten automatisiert über geeignete Protokolle, welche im Vorfeld zu bestimmen sind.

SICHERHEIT Der Sicherheitsaspekt bei der zwischenbetrieblichen Prozessausführung wird mit der Anwendung der Blockchain-Technologie als dezentrale Ausführungsplattform immer stärker in den Fokus gestellt [120]. Insbesondere die Eigenschaften der Blockchain als manipulationssicherer Datenspeicher werden dabei als wertvoll evaluiert. Im Speziellen bedeutet das, dass durch die Speicherung auf der Blockchain die Ausführungshistorie beziehungsweise die Prozesskontrolldaten im Nachhinein nicht mehr verändert werden können (Kapitel 4).

Im Gegensatz zu Blockchain-Protokollen als Implementierungsoption der dezentralen Kontrolle, wird bei einem verteilten System die Manipulationssicherheit nicht ohne weiteres unterstützt. So können zum Beispiel Zeitstempel von Nachrichten verfälscht werden, um zu vertuschen, dass die einzuhaltenden zeitlichen Bedingungen durch eigenes Verschulden verletzt worden sind. Auch kann die Verantwortlichkeit kritischer Aktivitäten nach deren Ausführung auf andere Teilnehmer übertragen werden, wenn innerhalb der Abarbeitung fahrlässig gehandelt wird und Schäden dadurch verursacht werden [169]. Derartige Konfliktsituationen können durch unterschiedliche Ereignisprotokolle entstehen, wenn die Teilnehmer die Prozesskontrolldaten beziehungsweise den erfolgten Nachrichtenaustausch unabhängig voneinander verwalten. Als Folge sollte bei sich misstrauenden Kollaborationspartnern eher die dezentrale Kontrolle mit manipulationssicheren Prozesskontrolldaten anstatt einem verteilten System Anwendung finden.

Auch bei einer zentralisierten Ausführung, verwaltet von einem der Kollaborationsteilnehmer oder einem externen Dienstleister, besteht die Möglichkeit, bestimmte Sicherheitsaspekte zu implementieren. Zum Beispiel kann ein Prozessteilnehmer einen Nachweis seiner Aktion erhalten, etwa in Form einer digitalen Signatur (Kapitel 4.2.2). Dies führt jedoch zu manuellem Implementierungsaufwand und außerdem hat immer noch eine einzelne Instanz die alleinige Kontrolle über alle Prozesskontrolldaten, während bei Blockchain-Systemen

die Daten repliziert verwaltet werden, was ein erfolgreiches Manipulieren der Daten erschwert.

Eine Analyse aller möglichen sicherheitsrelevanten Aspekte ist nicht das Ziel dieser Arbeit. Es sollte jedoch selbst die theoretische Möglichkeit der Manipulation explizit verhindern werden etwa durch dezentral verwaltete Prozesskontrolldaten in Blockchain-Systemen [115].

FAZIT Im Prozessmanagementbereich werden für zwischenbetriebliche Prozesse vor allem zwei Strategien für deren Umsetzung verfolgt. Auf der einen Seite beschreibt Weske et al. [187] Kollaborationen durch die Modellierung mittels (BPMN)-Choreografien, welche über das Prinzip der verteilten Kontrolle über lokal implementierte Prozesse umgesetzt werden. Diese Strategie ermöglicht die Koordination der Teilnehmer über einen festgelegten Nachrichtenaustausch und durch die verteilte Kontrolle und das abstrakte Nachrichtenaustauschmodell bleiben die Teilnehmer flexibel. Allerdings ist dabei keine Prozessausführung im Sinne von Kapitel 2 inklusive den genannten Vorteilen wie beispielsweise koordinierte Arbeitslisten möglich.

Eine Alternative zu BPMN-Choreografien für die Umsetzung zwischenbetrieblicher Prozesse bietet die dezentrale Kontrolle. Dabei wird ein global eindeutiger und akzeptierter Stand der Prozesskontrolldaten repliziert in einem dezentralen Netzwerk verwaltet [169]. Je nach verwendetem Synchronisierungsalgorithmus ist dieser auch robust gegenüber Manipulationsversuchen einzelner Teilnehmer (Kapitel 4). Die dezentrale Kontrolle basiert auf einem Prozessmodell und erlaubt damit auch die Implementierung der in Kapitel 2 vorgestellten Vorteile der Prozessausführungsdisziplin. In dieser Arbeit werden in Kapitel 5 und Kapitel 6 zwei Artefakte vorgestellt, wie die Prozessausführung mit dezentraler Kontrolle umgesetzt werden kann.

3.5 WORKFLOW-INTEROPERABILITÄT

Workflow Interoperabilität

Mit der Entwicklung des Internets wurde auch die systemgestützte zwischenbetriebliche Prozessausführung vorangetrieben. Im Jahr 1999 wurde eine der ersten Arbeiten in dieser Hinsicht veröffentlicht [2]. Darin sind am Beispiel von E-Commerce verschiedene Möglichkeiten diskutiert, wie zwischenbetriebliche Prozesse umgesetzt, also systemgestützt ausgeführt werden können. Die Arbeit konzentriert sich auf konzeptionelle Vorschläge zur Infrastruktur, abstrahiert jedoch von Modellierungsaspekten oder konkreten Implementierungen. Im Folgenden werden die Konzepte kurz vorgestellt und in Bezug zu den oben eingeführten Möglichkeiten der Umsetzung zwischenbetrieblicher Prozesse gesetzt.

3.5.1 *Verwandte Arbeiten*

In diesem Abschnitt werden die verwandten Arbeiten in Bezug auf die Formen der Workflow-Interoperabilität beschrieben. Eine umfassende Diskussion von verwandten Arbeiten im Bereich der Blockchain-basierten Prozessausführung erfolgt in Kapitel 8.

Capacity Sharing

Für den Einsatz des *Capacity Sharing*-Konzepts (deutsch: Gemeinsame Nutzung von Ressourcen) wird ein zentraler Koordinator des Prozesses angenommen, wohingegen die menschlichen Akteure unternehmensübergreifend verschiedenen organisationalen Entitäten zugeordnet sein können. Da darüber hinaus ein globaler Prozess angenommen wird, spiegelt diese Form die zentrale Ausführung eines Prozessmodells wider.

Chained Execution

Bei der *Chained Execution* (deutsch: verketteten Ausführung) wird der zwischenbetriebliche Prozess in sequenzielle Subprozesse zerlegt, welche dann autark der Reihe nach von den jeweiligen Teilnehmern bearbeitet werden.

Subcontracting

Die *Subcontracting*-Interoperabilitätsform (deutsch: vertragliche Delegation) nimmt einen administrierenden Kollaborationsteilnehmer an, welcher den zwischenbetrieblichen Prozess organisiert und bestimmte Aktivitäten gegebenenfalls als Subprozesse von anderen Teilnehmern abarbeiten lässt. Dabei wird im Gegensatz zu dieser Arbeit keine Gleichstellung aller Kollaborationsteilnehmer angenommen.

Case Transfer

Beim *Case Transfer* teilen sich alle Kollaborationsteilnehmer das gleiche Prozessmodell. Während der Ausführung wird die komplette Prozessinstanz jeweils an den nächsten Teilnehmer weitergegeben, welcher diese Instanz in das lokale WFMS lädt und die für ihn relevanten Teilaufgaben abarbeitet. Diese Form kann der verteilten Ausführung eines Prozessmodells zugeordnet werden.

Extended Capacity Sharing

Die Erweiterung *Extended Case Transfer* erlaubt darüber hinaus lokale Abweichungen der Prozessmodelle, wobei auf eine konforme Abbildung auf die globale Prozessbeschreibung geachtet werden muss.

Loosely Coupled Workflows

Die *Loosely Coupled Workflows* (deutsch: lose verknüpfte Prozesse) verwenden im Gegensatz zu den anderen Formen der Interoperabilität keine globale Prozessspezifikation des zwischenbetrieblichen Prozesses. Stattdessen wird der zwischenbetriebliche Prozess in für die jeweils beteiligten Organisationen relevanten Subprozesse aufgeteilt. Dies entspricht dem Vorgehen der verteilten Ausführung nachrichtenbasierter Kollaborationsmodelle.

3.5.2 *Decentralized Control: Eine neue Form von Workflow-Interoperabilität*

Da sich die dezentrale Ausführung von Prozessmodellen nicht in diese bisher beschriebenen Formen der zwischenbetrieblichen Prozessausführung einordnen lässt, wird in [169] das Konzept der *Decentralized Workflows* vorgeschlagen. Die Artefakte, welche in Kapitel 5 und Kapitel 6 vorgestellt werden, lassen sich dieser Form zuordnen.

konzeptionelle Idee

Die konzeptionelle Idee ist im Rahmen dieses Kapitels bereits erör-

tert und soll hier noch einmal kurz zusammengefasst werden. Das Decentralized Control-Prinzip basiert auf der Interpretation eines Prozessmodells, welches den zwischenbetrieblichen Prozess beschreibt und dabei die Flexibilität der Integration verschiedener Prozessperspektiven erlaubt. Dies hat zur Folge, dass hinsichtlich der organisationalen Perspektive auch ein Organisationsmodell verarbeitet werden kann, welches alle potenziell an der Ausführung des zwischenbetrieblichen Prozessmodells beteiligten organisationalen Entitäten verwaltet. Dies umfasst insbesondere auch die flexible Definition beliebiger Organisationsstrukturen, so wie in Kapitel 2.5.3 eingeführt. Weiterhin sind auch Daten in der informationsorientierten Perspektive und automatisierte Aktivitäten in der operationalen Perspektive zu unterstützen. Die Ausführung auf einer dezentralen Infrastruktur erfolgt dabei in mehreren Phasen. Während eine Aktion zur Aktualisierung des Prozessstatus stets von einem bestimmten Teilnehmer ausgeht, muss diese Aktion zuerst global bestätigt werden, bevor sie Auswirkungen auf den lokalen Datenstand hat. Der Grund ist, dass potenziell eine konkurrierende Anfrage von einem anderen Teilnehmer nahezu zeitgleich initiiert wurde. In diesem Fall muss sich das dezentrale Netzwerk zuerst auf eine global einheitliche Reihenfolge aller auftretenden Ereignisse einigen. Sobald eine globale akzeptierte Reihenfolge angenommen werden kann, können die Teilnehmer die entsprechenden Prozessaktionen in ihren lokalen WFMSs und Datenbeständen interpretieren. Aufgrund der Synchronisierung vorab wird so jeder Teilnehmer zur gleichen Zeit stets denselben Prozessstatus ableiten können, was schließlich eine dezentrale Ausführung des Prozesses ermöglicht.

Implementierungsaspekte

Von der konzeptionellen Idee sind die Implementierungsaspekte von Decentralized Control zu unterscheiden. Diese umfassen insbesondere Sicherheitsaspekte, zum Beispiel fehlerhafte und manipulierende Nachrichten. Außerdem muss im Rahmen einer Benutzerverwaltung sichergestellt sein, dass alle Benutzer und Initiatoren von Prozessaktionen sicher identifiziert werden können. Mögliche Angriffsvektoren, zum Beispiel, dass sich Benutzer als andere Benutzter ausgeben oder dass Ereignisprotokolle erfolgreich manipuliert werden können, sind explizit implementierungsseitig zu verhindern und werden nicht in die konzeptionelle Beschreibung von Decentralized Control aufgenommen, da möglicherweise unterschiedliche Kollaborationen unterschiedliche Sicherheitsvorgaben umsetzen möchten. Kapitel 5 stellt dabei ein Konzept zur Implementierung von Decentralized Control auf Basis des Ethereum-Protokolls vor, welches in Kapitel 4 eingeführt wird. Kapitel 6 stellt daraufhin ein Rahmenwerk zur flexibleren Implementierung von Decentralized Control vor, wodurch Systeme für die lokale Interpretation von Prozessen und für die Kommunikation und Synchronisierung von Nachrichten flexibel konfiguriert werden können.

Teil II

DEZENTRALE PROZESSAUSFÜHRUNG AUF DER BLOCKCHAIN

4

BLOCKCHAIN-BASIERTE SYSTEME

In Kapitel 3 werden System- und Netzwerkarchitekturen zur Umsetzung von zwischenbetrieblichen Prozessen eingeführt. Dafür existieren einerseits etablierte Ausführungsinfrastrukturen wie die Verwendung einer zentral verwalteten Infrastruktur oder die über ein Nachrichtenaustauschprotokoll koordinierten, lose gekoppelten Prozesse in einem verteilten System. Andererseits werden die durch fehlertolerante Synchronisierungsalgorithmen gestützten dezentralen Netzwerke unter dem Begriff von Decentralized Control als neuartige Möglichkeit für die zwischenbetriebliche Prozessausführung vorgestellt. Die ersten konkreten Ansätze im Bereich von Decentralized Control basieren dabei auf Blockchain-basierten Systemen [113, 170]. In Kapitel 5 wird beschrieben, wie sich auf Basis der Ethereum-Blockchain eine sichere, nicht manipulationsanfällige Ausführungsplattform für eine zwischenbetriebliche Prozessausführung umsetzen lässt. Zuvor werden in diesem Kapitel die Grundlagen von Blockchain-basierten Systemen vorgestellt. Dabei werden die Konzepte so weit wie möglich konzeptionell unabhängig von einem bestimmten Blockchain-Protokoll betrachtet. Aufgrund der Verwendung der Ethereum-Blockchain in Kapitel 5 beziehen sich die detaillierteren Informationen auf dieses spezielle Protokoll.

Im Folgenden führt Kapitel 4.1 die Blockchain als Black Box ein und zeigt dabei verschiedene Interaktionen mit einem Blockchain-System auf. Kapitel 4.2 beschreibt anschließend die grundlegenden Funktionsbausteine von Blockchain-Systemen, inklusive deren historische Entwicklung und Zusammensetzung. Im Anschluss erläutert Kapitel 4.3 die genaue Funktionsweise der intrinsischen Prozesse einer Blockchain, woraufhin Kapitel 4.4 eine abschließende Bewertung und Zusammenfassung dieses Kapitels, insbesondere vor dem Hintergrund der Prozessausführung liefert.

Es sei darauf hingewiesen, dass die Blockchain-Protokolle stetig weiterentwickelt werden und aktuelle Versionen der Protokolle sich möglicherweise von den hier beschriebenen Konzepten im Detail unterscheiden. Etwaige Diskrepanzen sollten jedoch keine Auswirkung für das Verständnis dieser Arbeit haben.

4.1 EINFÜHRUNG

Blockchain als komplexes Ökosystem

Die Blockchain-Technologie ist ein innovatives und komplexes Ökosystem, welches eine Art der Zusammenarbeit von beliebigen, potenziell unbekannten Teilnehmern erlaubt. Diese Eigenschaft ermöglicht unter anderem auch den Einsatz als Basisinfrastruktur bei zwischenbetrieblichen Kollaborationen [120].

Die Vielzahl an ineinandergreifenden Datenstrukturen und Algorithmen eines Blockchain-Protokolls erschweren eine geradlinige Einführung in derartige Systeme. Aus diesem Grund untergliedert dieses Kapitel die Einführung und stellt zunächst die Blockchain in Kapitel 4.1.1 als Black Box unter Vernachlässigung des inneren Aufbaus vor. Anschließend wird in Kapitel 4.1.2 die Blockchain für eine detailliertere Beschreibung aus verschiedenen Perspektiven beleuchtet. Abschließend folgt eine kurze Zusammenfassung der Einführung in Kapitel 4.1.3.

4.1.1 *Blockchain als Black Box*

Dieses Kapitel demonstriert die Interaktion von Benutzern mit einer Blockchain, ohne auf deren innere Strukturen näher einzugehen. Als Beispiel für die Erläuterungen dient das Prinzip einer *Kryptowährung*, einer rein digitalen und ausschließlich autonom vom dezentralen Netzwerk verwalteten Währung, welche durch das erste Blockchain-Protokoll namens Bitcoin in dieser Form erstmalig vorgestellt wurde [133].

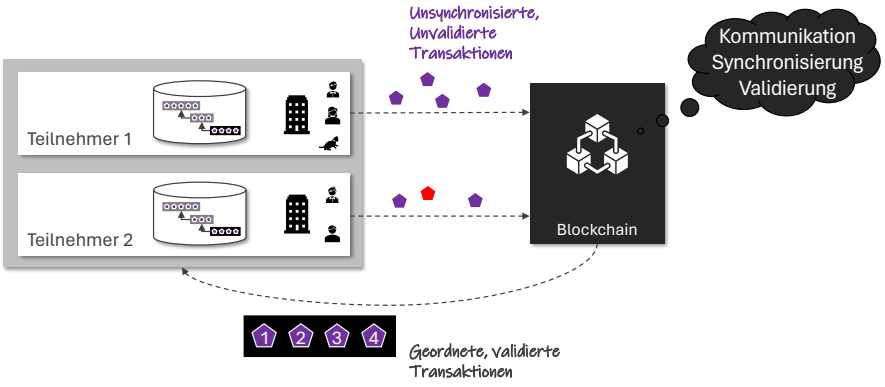

Abbildung 24: Blockchain-Systeme als Black Box

Transaktionen

Die Hauptfunktionalität der Black Box ist die Speicherung von Daten, welche von den Teilnehmern in Form von Transaktionen in das Blockchain-Netzwerk geschickt werden (lila Pentagone in Abbildung 24) und darüber auch aktualisiert werden können. Eine Transaktion in Bitcoin stellt eine Überweisung von Kryptowährung von einem Benutzerkonto auf ein anderes Benutzerkonto dar und wird über das Netzwerk an alle Teilnehmer verteilt. Dabei sind zwei Eigen-

schaften zu betrachten. Erstens ist in derartigen verteilten, asynchronen Kommunikationssystemen die Definition einer global eindeutigen Uhr nicht trivial [97], weswegen die Nachrichten beziehungsweise Transaktionen zu diesem Zeitpunkt keinen global akzeptierten Zeitstempel und keine global eindeutige Reihenfolge besitzen. Zweitens sind Betrugsversuche oder technische Fehler nicht auszuschließen, weswegen möglicherweise auch invalide Transaktionen in das Netzwerk geschickt werden. Die Validität der Transaktionen kann nach global akzeptierten Regeln geprüft werden, welche unter anderem im Blockchain-Protokoll selbst spezifiziert sind.

Hinzufügen neuer Daten

Das Hinzufügen neuer Daten beziehungsweise neuer Transaktionen erfolgt nicht kontinuierlich, sondern in Stapeln. Dabei sammelt die Blockchain alle Transaktionen und produziert in einem festen Zeitintervall einen neuen sogenannten *Block*, welcher einen Teil der gesammelten Transaktionen enthält (schwarzes Rechteck unten in Abbildung 24). Dabei werden einerseits nur valide Transaktionen inkludiert und zum anderen sind die Transaktionen innerhalb eines Blocks eindeutig geordnet [190, Kap. 4.3]. Da ein neuer Block stets eine eindeutige Referenz auf seinen Vorgänger hält, sind auch die Blöcke selbst in einer geordneten Liste verwaltet, sodass insgesamt in der Blockchain-Datenstruktur alle Transaktionen eine global eindeutige Reihenfolge besitzen [190, Kap. 4.3]. Auf diese Weise einigt sich das gesamte Netzwerk in der Regel auf einen äquivalenten Datenbestand. Die Synchronisierung der Reihenfolge aller Transaktionen ist insofern wichtig, als dass eine Transaktion von einer vorherigen Transaktion invalidiert werden könnte, wenn nach der ersten Transaktion dem sendenden Benutzerkonto keine ausreichende Menge an Kryptowährung mehr zugeordnet ist, um die zweite Überweisung zu tätigen. Über die Blockchain werden zusammengefasst die folgenden Punkte umgesetzt: die Kommunikation, also die Verteilung aller Transaktionen und neuer Blöcke, die Synchronisierung, also das Ordnen von Transaktionen, und die Validierung der Transaktionen nach bestimmten Regeln [190, Kap. 6].

Überweisung von Kryptowährung

Aus technischer Sicht erfolgt die Interaktion mit einer Blockchain über einen *Account* (deutsch: Benutzerkonto), welcher mithilfe von bestimmten Anwendungen erstellt werden kann.[1] Jeder Account ist im Grunde ein Schlüsselpaar eines asymmetrischen Kryptosystems [8, Kap. 5], bestehend aus einem öffentlichen und einem geheimen Schlüssel (Kapitel 4.2.2).

Dem Account werden, ähnlich wie bei einem Bankkonto, über den öffentlichen Schlüssel eine Reihe von monetären Einheiten (*Kryptowährung*) zugeordnet. Über den geheimen Schlüssel kann der Benutzer die Einheiten *entsperren* und so an einen anderen Account *überweisen*. Dies geschieht prinzipiell über die Strukturierung der erforderlichen Informationen in einer Transaktion, welche anschließend aus Sicher-

1 https://github.com/ethereum/go-ethereum, besucht am 26.11.2023

heitsgründen mit dem privaten Schlüssel des Senders signiert wird. Alle Überweisungen sind demnach ähnlich einem verteilten Kassenbuch (englisch: *distributed ledger*) in den Transaktionen in den Blöcken auf der Blockchain gespeichert, wodurch stets geprüft werden kann, ob einem Account noch genügend Kryptowährung zugeordnet ist, um eine bestimmte Überweisung auszuführen. Diese Überprüfung erfolgt im Zuge der Validierung der Transaktionen.

weiterführende Anwendung in Smart Contracts

Im vorherigen Abschnitt wird die grundsätzliche Interaktion mit der Blockchain aus Nutzerperspektive unter Verwendung des Anwendungsfalls einer Überweisung an Kryptowährung erklärt. Im weiteren Verlauf wird gezeigt, dass den Transaktionen auch ausführbarer Quelltext zugeordnet werden kann, welcher wiederum durch weitere Transaktionen aufgerufen werden kann. Über diese kleinen Programme, welche als *Smart Contracts* bezeichnet werden, kann die Validität von Transaktionen beliebig komplex spezifiziert werden [190, Kap. 9], was unter anderem zur Ausführung von Geschäftsprozessen genutzt wird [170]. Eine Transaktion ist demnach nur valide, wenn die Ausführung der korrespondierenden Smart Contract-Funktion fehlerfrei abläuft, wobei dann Transaktionen, welche invalide Prozessaktionen referenzieren, einen Fehler verursachen und so nicht in einen Block aufgenommen werden. Im Kapitel 4.2 wird gezeigt, welche Datenstrukturen und Algorithmen dabei hauptverantwortlich für einen ordnungsgemäßen und vor allem sicheren Betrieb einer Blockchain sind.

4.1.2 *Die Blockchain aus unterschiedlichen Perspektiven*

Die Blockchain-Technologie ist ein vielschichtiges System und kann unter anderem als Netzwerk, Datenspeicher oder als vertrauenswürdige Instanz bezeichnet werden [174]. Der folgende Abschnitt beleuchtet Blockchain-Systeme aus verschiedenen Perspektiven.

Peer-to-Peer-Netzwerk

NETZWERKPERSPEKTIVE Die zugrundeliegende Infrastruktur einer Blockchain ist ein Peer-to-Peer-Netzwerk, über das alle Informationen wie etwa die Transaktionen ausgetauscht werden. In einem Peer-to-Peer-Netzwerk sind alle verbundenen Knoten gleichartig, während im Gegensatz dazu bei einer Client-Server-Architektur der Server eine Anwendung oder eine Datenmenge zentral verwaltet. Durch die Peer-to-Peer-Netzwerkarchitektur wird eine dezentrale Kontrolle der Daten über automatisierte Algorithmen ermöglicht. Die Verteilung von Transaktionen erfolgt ebenfalls automatisiert über einen *Flooding*-Algorithmus (deutsch: Flutalgorithmus) [83]. Aus technischer Sicht erfolgt die Teilnahme in einem Blockchain-Netzwerk üblicherweise über die Installation und Ausführung einer Client-Anwendung, welche sich mit bestehenden Knoten des Netzwerks verbindet.[2] Grund-

2 https://github.com/ethereum/go-ethereum, besucht am 06.02.2024

sätzlich kann jeder über eine Client-Anwendung einem öffentlichen Blockchain-Netzwerk beitreten. Manche Blockchain-Protokolle ermöglichen auch die Limitierung der Sichtbarkeit des Zugriffs, um private Netzwerke mit kontrollierter Teilnahme von nur bekannten Knoten zu erstellen [174]. Im Geschäftsprozessmanagementbereich kann dies dazu genutzt werden, um die Offenlegung von sensiblen Daten einzuschränken [120].

verkettete Datenstruktur

DATENPERSPEKTIVE Der Begriff *Blockchain* bezeichnet per se eine verkettete Datenstruktur, in der alle Transaktionen in Blöcken gruppiert verwaltet werden [190, Kap. 2]. Neben den Transaktionen besitzt jeder Block auch einen *(Block)-Header*. Darin referenziert ein nachfolgender Block stets seinen direkten Vorgänger. Diese Datenstruktur mit der gesamten Historie an Transaktionen wird prinzipiell auf jedem Knoten im Netzwerk gespeichert, wodurch jeder Knoten auch neue Transaktionen validieren kann. Neu validierte Transaktionen werden immer im Rahmen eines neuen Blocks an die bestehende Datenstruktur angehängt. Durch gewisse Prinzipien, unter anderem durch sogenannte Konsensmechanismen, wird sichergestellt, dass sich alle den Regeln folgenden Knoten im Netzwerk immer nach einer gewissen Zeit auf den gleichen Datenstand einigen [100].

Robustheit gegen Manipulation

INTEGRITÄTSPERSPEKTIVE Über die Netzwerk- und Datenperspektive wird eine verkettete, stetig wachsende Datenstruktur repliziert auf verschiedenen Knoten in einem Peer-to-Peer-Netzwerk verwaltet. Mit Millionen von verwalteten Transaktionen[3] und mehreren 100 GB an Daten[4] sind effiziente Maßnahmen hinsichtlich der Integrität der Datenstruktur zu ergreifen. Mit der Integritätsperspektive wird beschrieben, dass erfolgreich manipulierende Änderungen an der Datenstruktur wie das unerlaubte Einfügen, Ändern oder Löschen von Transaktionen oder Blöcken durch eine effiziente Integritätsprüfung verhindert werden können.

Verknüpfung der Header

Die Integrität kann durch die spezielle Verknüpfung der Blöcke in der Blockchain gewährleistet und effizient geprüft werden. In jedem Header findet sich eine prüfsummenartige Zusammenfassung aller Transaktionen des entsprechenden Blocks [190, Kap. 4]. Außerdem beeinflusst weiterhin der Header eines Blocks direkt die Berechnung des Headers des nächsten Blocks. Somit wird eine Prüfsumme der gesamten Datenstruktur verwaltet, welche mit jedem neuen Block neu berechnet wird. Wenn eine Transaktion in einem historischen Block also modifiziert wird, ändert sich der Header dieses Blocks und damit alle Header der darauffolgenden Blöcke.

3 https://etherscan.io/chart/tx, besucht am 06.02.2024
4 https://etherscan.io/chart/blocksize, besucht am 06.02.2024

Beispiel: Integritätsprüfung. Sei der Header von Block b_{i-1} gleich dem Wert 3. Sei weiterhin angenommen, dass die Transaktionen tx_1 und tx_2 im Block b_i die Prüfsummen 1 und 4 aufweisen. Die Prüfsumme aller Transaktionen im Block b_i kann beispielsweise über die Summe aller Prüfsummen der einzelnen Transaktionen gebildet werden: $1 + 4 = 5$. Die Prüfsumme im Header von Block b_i kann nun aus der Summe der eben berechneten Prüfsumme und dem Header des Vorgängerblocks b_{i-1} berechnet werden: $5 + 3 = 8$.

Wenn ein Angreifer eine Transaktion selbst oder die Menge an gespeicherten Transaktionen manipuliert, ändert sich damit der Header des entsprechenden Blocks. Durch die Verkettung der Blöcke über den Header propagiert sich diese Änderung bis zum aktuellsten Block. Für die Etablierung der Manipulation muss der Angreifer nun alle Block-Header neu berechnen, da das Netzwerk stets die längste Kette mit den meisten Blöcken als die gültige annimmt. Wie in Kapitel 4.2.4 beschrieben ist die Berechnung jedoch unpraktikabel aufwendig. Somit sind die Datenstruktur zusammen mit innovativen Algorithmen zur Block-Header-Berechnung essenzielle Bausteine für die Integrität der verwalteten Daten, was in Kapitel 4.3 vertieft wird. Die tatsächlich im Protokoll verwendeten Datenstrukturen sind in der Regel komplexer und hier für eine anschauliche Erläuterung der Konzepte vereinfacht dargestellt. Für eine genauere Beschreibung sei auf die offizielle technische Beschreibung verwiesen [190].

Bedeutung für die Prozessausführung

Die Integrität der Datenstruktur ist im Allgemeinen eine essenzielle nichtfunktionale Anforderung. Auch in Hinblick auf die Prozessausführung werden alle Aktionen über Transaktionen abgebildet, das heißt, dass auch das Ereignisprotokoll im Nachhinein nicht verändert werden kann. Somit können keine Aktivitäten hinzugefügt oder herausgelöscht oder Datenwerte oder verantwortliche Personen nicht verändert werden.

OPERATIONALE PERSPEKTIVE In Blockchain-Systemen können einerseits unbeschränkt viele und andererseits pseudonymisierte Teilnehmer im Netzwerk einen Account erstellen, wobei folglich die Erstellung mehrerer Accounts von einem einzigen Teilnehmer nicht ausgeschlossen ist. Aus diesem Grund sind bestimmte Sicherheitsaspekte in Bezug auf den ordnungsgemäßen operativen Betrieb zu berücksichtigen.

Blockerstellung

Einen essenziellen Beitrag zur Aufrechterhaltung der Funktionalität von Blockchains liefert die Blockerstellung. Für die Erstellung eines neuen Blocks und damit für die Auswahl und die Ordnung valider Transaktionen wird ein Teilnehmer beauftragt, der über ein bestimmtes Auswahlverfahren bestimmt wird. Dabei sind unter anderem die folgenden Aspekte zu berücksichtigen. Wird ein Teilneh-

mer zufällig selektiert, können Teilnehmer durch die mehrfache Erstellung von Accounts die Wahrscheinlichkeit erhöhen, gewählt zu werden und sich dadurch einen unfairen Vorteil verschaffen (Sybil Attack) [60]. Diese Herausforderung lösen Blockchain-Protokolle beispielsweise dadurch, dass die Auswahl auf Basis der verfügbaren Rechenkapazität (*Proof of Work*, Kapitel 4.2.4) oder der einem Account zugewiesenen Menge an Kryptowährung (*Proof of Stake*) getroffen wird [19].

Konformität

Ein weiterer Aspekt betrifft den nächsten Schritt, nämlich wenn ein ausgewählter Knoten invalide Blöcke beziehungsweise Blöcke mit invaliden Transaktionen propagiert. Dieser Angriffsvektor wird durch die Pseudonymität der Teilnehmer begünstigt und stört den Betrieb des Netzwerks im Stil eines *Denial-of-Service*-Angriffs. Eine Lösung kann auch hier auf Basis von Kryptowährungen umgesetzt werden, indem ein Blockersteller in monetärer Hinsicht entlohnt wird [133, Kap. 6]. Beim Proof of Work-Verfahren verursacht etwa die Erstellung eines validen Blocks aufgrund eines hohen erforderlichen Rechenaufwands erhebliche Kosten. Die Blockersteller werden im Gegenzug in Form von Kryptowährung belohnt [7, S. 223], allerdings nur bei Inklusion des vorgeschlagenen Blocks in die Blockchain, wenn alle Netzwerkteilnehmer den erstellen Block als valide betrachten. Bei privaten Netzwerken mit beschränkter Teilnehmerzahl können anstelle von Proof of Work auch vertrauenswürdige Teilnehmer spezifiziert werden, aus denen über ein herkömmliches Auswahlverfahren, etwa eine rotationsbasierte Auswahl, der nächste Blockersteller erkoren wird [150].

Non-Repudiation

Ein weiteres Sicherheitskriterium von Blockchain-Systemen ist die Nichtabstreitbarkeit (englisch: *non-repudiation*) [47]. Diese besagt, dass der Sender einer Transaktion im Nachhinein nicht abstreiten kann, der tatsächliche Sender zu sein. Andernfalls könnte bei der Prozessausführung eine verantwortliche Person leugnen, eine bestimmte Aktivität ausgeführt zu haben. Die Nichtabstreitbarkeit kann mittels digitaler Signaturen implementiert werden (Kapitel 4.2.2).

Weitere Sicherheitsanforderungen

Es existieren weitere Angriffsvektoren und entsprechende Sicherheitsmechanismen von Blockchain-Systemen [102]. Diese tangieren allerdings weniger die hier diskutierte Entwicklung der Artefakte zur Prozessausführung, sondern primär die Weiterentwicklung und Absicherung der Blockchain-Protokolle per se. Der Fokus dieser Arbeit ist keine Analyse der Anwendbarkeit der Blockchain-basierten Artefakte hinsichtlich Sicherheitskriterien. Stattdessen sollen einerseits bereits existierende Artefakte zur Prozessausführung speziell für das Ethereum-Protokoll in Kapitel 5 weiterentwickelt werden und andererseits soll in Kapitel 6 ein Framework zur flexiblen Austauschbarkeit von Blockchain-Protokollen und ähnlichen Synchronisierungsalgorithmen neu entwickelt werden. Aus diesen Gründen erfolgt hier keine tiefere Sicherheitsanalyse der Blockchain-Protokolle.

dezentrale Anwendungen über Smart Contracts

ANWENDUNGSPERSPEKTIVE Fortschrittliche Blockchain-Protokolle wie Ethereum erlauben eine flexible Definition der Validierungsregeln von Transaktionen. Diese Regeln können programmatisch in Smart Contracts implementiert und ebenfalls über Transaktionen in der Blockchain persistiert werden. Smart Contracts sind konzeptionell kleine Programme, welche Variablenfelder und Funktionen definieren können [8, Kap. 7]. Die Funktionen können dann über das Senden von Transaktionen aufgerufen werden. Zum Beispiel können Smart Contracts in den Variablen die Struktur und den Fortschritt eines Prozessmodells speichern und Funktionen bereitstellen, um einen Prozessstatus zu aktualisieren [170]. Dies erlaubt die Umsetzung komplexer Anwendungen, sodass auch die Funktionalitäten eines WFMS auf Basis eines Blockchain-Systems bereitgestellt werden können [113, 168].

4.1.3 *Zusammenfassung: Blockchain*

Zusammenfassend lässt sich ein Blockchain-System als Plattform betrachten, welches dezentrale Anwendungen zusammen mit einer dezentralen Datenhaltung ermöglicht und dabei durch geeignete Konsensverfahren wie Proof of Work eine hochskalierbare Menge von pseudonymisierten Benutzerkonten erlaubt. Anstatt, dass eine einzige Entität die Infrastruktur verwaltet und kontrolliert, sind alle Daten und Programme repliziert auf den verschiedenen Knoten im Netzwerk verteilt. Geeignete Synchronisierungsalgorithmen sorgen dafür, dass sich alle dem Protokoll folgenden Knoten auf denselben Datenstand einigen können. Dabei sind Blockchain-Systeme robust gegenüber einem bestimmten Anteil an technisch fehlerhaften oder bewusst manipulierenden Teilnehmern.

Blockchain-Systeme können auch mit traditionelleren Synchronisierungsalgorithmen konfiguriert werden, wonach nur eine ausgewählte, vertrauenswürdige Menge an Teilnehmern das Netzwerk und die Daten verwalten und neue Blöcke vorschlagen dürfen. Dadurch kann einerseits den hohen Kosten von Proof of Work entgegnet werden, allerdings führt dies andererseits auch zu einer Zentralisierung des Netzwerks, was ein gewisses Maß an Vertrauen in die kontrollierenden Instanzen für den praktischen Einsatz erfordert.

4.2 GRUNDLEGENDE KONZEPTE VON BLOCKCHAIN-SYSTEMEN

Trotz der innovativen und disruptiven Eigenschaften von Blockchain-Protokollen basieren diese auf teils jahrzehntealten Konzepten. Erst die spezielle Zusammensetzung der Komponenten in den Systemen resultiert in den innovativen Eigenschaften. Dieses Kapitel stellt mit den Hash-Funktionen in Kapitel 4.2.1, digitalen Signaturen in Kapitel 4.2.2 und Merkle-Bäumen in Kapitel 4.2.3 zuerst die wichtigsten

zugrundeliegenden Konzepte kurz vor. Anschließend werden die darauf basierenden weiterführenden Konzepte Proof of Work in Kapitel 4.2.4 und Smart Contracts in Kapitel 4.2.5 eingeführt. Schließlich wird deren Zusammenspiel in Zusammenhang mit der Funktionalität von Blockchain-Protokollen in Kapitel 4.3 erläutert. Die Bestandteile von Blockchain-Protokollen und deren Ursprung ist in Abbildung 25 skizziert, wobei die blau hinterlegten Ziffern die folgenden erklärenden oder verwendenden Kapitel referenzieren.

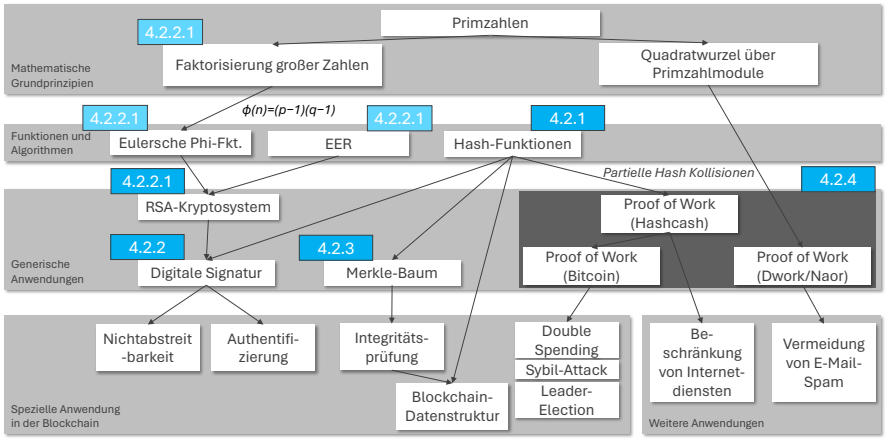

Abbildung 25: Übersicht über die Grundprinzipien hinter Blockchain-Systemen (Auswahl). EER: Erweiterter Euklidischer Algorithmus

Die folgenden Grundlagen insbesondere die historische Entwicklung der Konzepte ist für die Artefakte in Kapitel 5 und Kapitel 6 weniger relevant. Stattdessen dienen die Informationen einem weiterführenden Verständnis für eine mögliche Weiterentwicklung der Artefakte in künftigen Arbeiten.

4.2.1 *Hash-Funktionen*

Hash-Funktionen

Hash-Funktionen sind ein wichtiger Bestandteil von Blockchain-Protokollen und spielen eine große Rolle unter anderem bei der Erstellung digitaler Signaturen, Merkle-Bäumen und beim Konsensverfahren Proof of Work. Die folgenden Erläuterungen von Hash-Funktionen sind angelehnt an [121, Kap. 9].

Eine Hash-Funktion h ist eine mathematische Funktion, welche Eingaben beliebiger, endlicher Länge auf Werte mit konstanter Länge n abbildet.

$$h : D \to R, \quad |D| > |R|$$

nicht kollisionsfrei

Folglich sind Hash-Funktionen nicht kollisionsfrei, das heißt, es exis-

tieren zwei Eingaben d_1 und d_2, welche bei einer gegebenen Hash-Funktion h denselben Funktionswert r aufweisen.

$$\exists d_1,\ d_2 \mid h(d_1) = r \ \wedge \ h(d_2) = r \text{ mit } |r| = n$$

Des Weiteren gilt, dass der Funktionswert $h(d)$, auch Hashwert von d genannt, leicht zu berechnen ist, wenn die Eingabe d und die Funktion h bekannt sind.

kryptografische Hash-Funktionen

Bei sicherheitskritischen Anwendungen wie asymmetrischen Kryptosystemen, welche zur Verschlüsselung oder dem digitalen Signieren von Daten eingesetzt werden, kommen sogenannte kryptografische Hash-Funktionen zum Einsatz. Diese haben die folgenden wichtigen zusätzlichen Eigenschaften.

Sei h nun eine kryptografische Hash-Funktion und d_1, d_2 zwei Elemente der Definitionsmenge mit $h(d_1) = r_1$, $h(d_2) = r_2$. Dann gilt:

PREIMAGE RESISTANCE Für einen gegebenen Funktionswert r_1 ist es praktisch unmöglich, einen Wert $d_1 \in D$ zu finden, sodass $h(d_1) = r_1$.

2ND-PREIMAGE RESISTANCE Für einen gegebenen Wert $d_1 \in D$ ist es praktisch unmöglich, einen zweiten Wert $d_2 \in D$ zu finden mit $d_2 \neq d_1$, sodass $h(d_1) = h(d_2)$.

COLLISION RESISTANCE Es ist praktisch unmöglich zwei verschiedene Werte d_1, $d_2 \in D$ zu finden, sodass $h(d_1) = h(d_2)$.

4.2.2 *Digitale Signaturen*

Ähnlich wie eine handschriftliche Unterschrift auf einem ausgedrucktem Dokument oder Vertrag, stellt eine digitale Signatur eines Nutzers einen sicheren Nachweis über die Kenntnisnahme oder Zustimmung des Signierenden dar. Die Ziele einer digitalen Signatur umfassen einerseits die eindeutige Bestimmung der Urheberschaft einer Nachricht oder eines Dokuments und andererseits die Integritätsprüfung einer Nachricht oder eines Dokuments. Voraussetzung dafür ist unter anderem, dass diese digitale Signatur eindeutig zuzuordnen und nicht abstreitbar ist.

symmetrische und assymmetrische Kryptosysteme

Fortgeschrittene Signaturschemata basieren auf sogenannten asymmetrischen Kryptosystemen, welche in den 1970er Jahren entwickelt wurden. Einer der ersten Meilensteine in diesem Bereich ist ein Protokoll zum sicheren Schlüsselaustausch für symmetrische Verschlüsselungsverfahren auf Basis eines (asymmetrischen) Public-Key-Verfahrens von Whitfield Diffie und Martin Hellman im Jahre 1976 [36].

Diffie-Hellmann-Protokoll

Bei symmetrischen Verschlüsselungsverfahren müssen beide Kommunikationspartner Kenntnis von einem identischen Schlüssel haben. Der Austausch dieses Schlüssels gilt als kritisch und ist ein potenzieller Angriffsvektor. Bei asymmetrischen Verschlüsselungen oder

Public-Key-Verfahren existiert für jeden Kommunikationspartner jeweils ein öffentlicher Schlüssel und ein privater oder geheimer Schlüssel, der nur jeweils dem sendenden Teilnehmer der Kommunikation bekannt sein muss. Beim Diffie-Hellmann-Protokoll können durch geschickte mathematische Berechnungen unter Verwendung eines gemeinsamen öffentlichen Schlüssels und des jeweiligen privaten Schlüssels der gleiche gemeinsame geheime Schlüssel für die anstehende symmetrische Verschlüsselung abgeleitet werden.

RSA-Kryptosystem

Basierend auf dem Diffie-Hellmann-Protokoll wird zwei Jahre später im Jahr 1978 von Rivest, Shamir und Adleman das *RSA-Kryptosystem* vorgestellt [146], welches sowohl zur Verschlüsselung von Nachrichten verwendet werden kann als auch zur Erzeugung digitaler Signaturen. Der folgende Abschnitt beschreibt die Funktionsweise der digitalen Signatur nach dem RSA-Kryptosystem und illustriert das Vorgehen anschließend an einem Beispiel.

Erstellung einer Digitalen Signatur nach dem RSA-Kryptosystem

Ein Dokument, welches Alice von Bob erwartet, soll hinsichtlich der folgenden Kriterien gesichert werden. Das Dokument könnte auch eine Transaktion in einem Blockchain-Netzwerk sein und damit die Information über eine Überweisung beinhalten, oder dass der Sender eine bestimmte Aktivität ausgeführt hat.

- Bob soll eindeutig als Verfasser und Sender des Dokuments nachvollzogen werden können.
- Die Integrität des Dokuments soll sichergestellt werden, das bedeutet, dass weder Bob noch Alice oder ein Angreifer das Dokument unbemerkt verändern können soll.

Alice und Bob benutzen für die Erstellung einer digitalen Signatur das RSA-Kryptosystem, um das zu sendende Dokument nach den oben genannten Kriterien zu sichern. Die einzelnen auszuführenden Schritte werden im Folgenden zuerst kurz zusammengefasst und anschließend in einem Beispiel angewandt.

1. Bob wählt zwei geheime Primzahlen p und q und berechnet das Produkt $n = p \cdot q$.
2. Bob wählt eine Zahl e, die teilerfremd zu $\phi(n) = (p-1) \cdot (q-1)$ ist, also $ggT(e, \phi(n)) = 1$.[5]
3. Bob berechnet d, sodass $d \cdot e \equiv 1 \bmod (p-1) \cdot (q-1)$ unter Verwendung des erweiterten euklidischen Algorithmus (siehe unten im Beispiel).

5 Die Eulersche Phi-Funktion $\phi(n)$ gibt die Anzahl der zu n teilerfremden Ganzzahlen an, welche kleiner als n sind. Wenn n das Produkt zweier Primzahlen p und q ist, gilt $\phi(n) = (p-1) \cdot (q-1)$.

4. Bob veröffentlicht n und e als öffentlichen Schlüssel und hält p, q und d als privaten Schlüssel geheim.
5. Für eine Nachricht m berechnet Bob die digitale Signatur s mit $s = m^d \bmod n$.
6. Alice kann die Signatur überprüfen, indem sie den Wert x berechnet mit $x = s^e \bmod n$. Wenn die Werte x und m übereinstimmen, gilt die Signatur als verifiziert.

Das RSA-Kryptosystem basiert unter anderem auf der Komplexität, die Zahl $n = p \cdot q$ ohne Kenntnis von p und q zu faktorisieren, was ein grundsätzliches Problem in der Zahlentheorie darstellt [35].

Beispiel: Erzeugung und Prüfung einer Signatur nach dem RSA-Kryptosystem.

1. Bob wählt für die geheimen Primzahlen $p = 3$ und $q = 11$ und berechnet das Produkt:

   $$n = p \cdot q = 3 \cdot 11 = 33$$

2. Damit kann der Funktionswert der eulerschen Phi-Funktion wie folgt berechnet werden:

   $$\phi(33) = (p-1) \cdot (q-1) = 2 \cdot 10 = 20$$

   Bob wählt nun den öffentlichen Exponenten $e = 3$ als eine teilerfremde Zahl zu $\phi(33) = 20$, da $ggT(3, 20) = 1$.

3. Über den erweiterten euklidischen Algorithmus bestimmt Bob anschließend den geheimen Exponenten (privaten Schlüssel) $d = 7$ als multiplikativ inverses Element, das heißt, es gilt $(d \cdot e) \mod \phi(n) = 1$ beziehungsweise $d \cdot e \equiv 1 \bmod \phi(n)$.

   Die folgenden Ausführungen für die Durchführung des erweiterten euklidischen Algorithmus basieren auf der Erklärung in [26, Seite 16]. Der euklidische Algorithmus dient dazu, den größten gemeinsamen Teiler zweier Zahlen zu ermitteln, wohingegen der erweiterte euklidische Algorithmus zusätzlich den größten gemeinsamen Teiler als Linearkombination der beiden Zahlen darstellt. Das heißt, für zwei nicht-negative Zahlen a und b werden die Werte u, v und g berechnet, sodass $a \cdot u + b \cdot v = g$, wobei g der größte gemeinsame Teiler von a und b ist. Aus der Linearkombination lässt sich aus v der Wert für den geheimen Schlüssel d ableiten.

   Im Folgenden wird der Algorithmus für die Werte $a \leftarrow \phi(n) = 20$ und $b \leftarrow e = 3$ durchgeführt. Zu Beginn werden die Werte $u \leftarrow 1$, $g \leftarrow a$, $v_1 \leftarrow 0$, $v_3 \leftarrow b$ initialisiert. Da die Abbruchbedingung $v_3 = 0$ noch nicht erfüllt ist, werden die folgenden Berechnungsvorschriften ausgeführt:

- $q \leftarrow \lfloor \frac{g}{v_3} \rfloor$
- $t_3 \leftarrow g \bmod v_3$
- $t_1 \leftarrow u - q \cdot v_1$

Anschließend werden folgendermaßen die Werte aktualisiert:

- $u \leftarrow v_1$
- $g \leftarrow v_3$
- $v_1 \leftarrow t_1$
- $v_3 \leftarrow t_3$

Die einzelnen Berechnungsschritte sind in Tabelle 3 aufgelistet. In der letzten Zeile ist die Abbruchbedingung $v_3 = 0$ erfüllt. Aus der Gleichung $a \cdot u + b \cdot v = g$ lässt sich nun mit den Eingabewerten $a = \phi(n) = 20$ und $b = e = 3$ sowie mit den Werten $u = -1$ und $g = 1$ aus der Tabelle 3 der Wert v berechnen:

$$v = \frac{g - a \cdot u}{b} = \frac{1 - 20 \cdot (-1)}{3} = \frac{21}{3} = 7$$

4. Damit hat Bob ein valides Schlüsselpaar erstellt:
   öffentlicher Schlüssel: $n = 33$, $e = 3$;
   geheimer Schlüssel: $d = 7$ ($p = 3$, $q = 11$)

5. Mit diesen Informationen kann Bob nun eine Signatur für eine Nachricht m erstellen. Der private Schlüssel d wird von Bob für die Signaturerzeugung verwendet, sodass Alice die Signatur mit dem öffentlichen Schlüssel von Bob (e, n) verifizieren kann. Im folgenden Beispiel wird eine Signatur s für die Nachricht $m = 17$ erstellt. Dafür berechnet Bob $s = m^d \bmod n = 17^7 \bmod 33 = 410338673 \bmod 33 = 8$. Die Signatur s für die Nachricht $m = 17$ ist damit der Wert 8.

6. Mithilfe des öffentlichen Exponenten $e = 3$ kann jetzt die Signatur überprüft werden und damit die Autorenschaft der Nachricht verifiziert werden. Für die Verifizierung berechnet Alice $x = s^e \bmod n = 8^3 \bmod 33 = 512 \bmod 33 = 17$.

   Wegen $x = m$, kann angenommen werden, dass die Signatur der Nachricht von derjenigen Person erstellt wurde, die den geheimen Schlüssel zu dem öffentlichen Exponenten e kennt.

4.2.3 *Merkle-Baum*

Merkle-Baum

Der Merkle-Baum (englisch: *Merkle tree*) ist eine Datenstruktur, welche von Ralph Merkle entwickelt und mit seiner Dissertation 1979

u	g	v_1	v_3	q	t_3	t_1
1	20	0	3	$\lfloor \frac{20}{3} \rfloor = 6$	$20 \bmod 3 = 2$	$1 - 6 \cdot 0 = 1$
0	3	1	2	$\lfloor \frac{3}{2} \rfloor = 1$	$3 \bmod 2 = 1$	$0 - 1 \cdot 1 = -1$
1	2	-1	1	$\lfloor \frac{2}{1} \rfloor = 2$	$2 \bmod 1 = 0$	$1 - 2 \cdot (-1) = 3$
-1	1	3	0			

Tabelle 3: Erweiterter euklidischer Algorithmus nach [26, Seite 16]

veröffentlicht wurde [123]. Ein Merkle-Baum verwaltet Daten in einer Baumstruktur und erlaubt einerseits eine effiziente Integritätsprüfung von Daten und andererseits lässt sich überprüfen, ob ein Datensatz ein bestimmtes Element enthält, ohne den gesamten Datensatz durchsuchen zu müssen. Merkle-Bäume liefern damit einen entscheidenden Beitrag bei der Integritätsprüfung in Blockchain-Systemen. Die in Kapitel 4.2.1 eingeführten Hash-Funktionen spielen bei der Konstruktion und Verwendung von Merkle-Bäumen eine essenzielle Rolle. Auf eine formale Definition eines Merkle-Baums wird verzichtet und die Datenstruktur stattdessen exemplarisch eingeführt.

4.2.3.1 *Aufbau und Konstruktion eines Merkle-Baums*

Konstruktion eines Merkle-Baums

Für die Konstruktion des Merkle-Baums in Abbildung 26 wird die Hash-Funktion $h(x) = x \bmod 10$ verwendet, welche für jede ganzzahlige Eingabe, den Wert modulo 10 ausgibt. Weiterhin wird die Zeichenfolge `H, A, S, E, L, O, F, F` beziehungsweise die zugehörigen *ASCII*-Werte `72, 65, 83, 69, 76, 79, 70, 70` als Datengrundlage angenommen. Diese Datensätze bilden die Blätter des Merkle-Baums, also die Knoten auf der tiefsten Ebene des Baumes, welche keine Kindknoten besitzen. Die Elternknoten jeweils zweier Kindknoten werden durch die sequentielle Anwendung einer Additionsoperation beziehungsweise einer Konkatenationsfunktion bei Zeichenketten und der Hash-Funktion h berechnet.

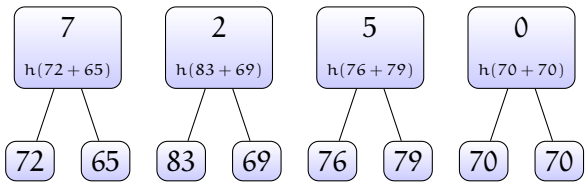

Abbildung 26: Konstruktion des Merkle-Baums

Dieses Verfahren wird angewandt bis der einzige Wurzelknoten des Baumes gefunden und berechnet ist. Wenn die Anzahl der Datenelemente keiner Zweierpotenz entspricht, werden die fehlenden Elemente durch Surrogate ergänzt, um eine vollständige binäre Baumstruktur zu gewährleisten.

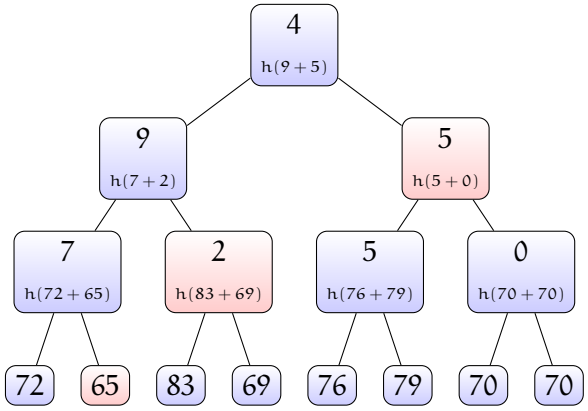

Abbildung 27: Vollständiger Merkle-Baum

4.2.3.2 *Anwendung des Merkle-Baums*

praktische Anwendung des Merkle-Baums

In diesem Abschnitt wird anhand des Merkle-Baums in Abbildung 27 gezeigt, wie ein Existenzbeweis oder eine Integritätsüberprüfung effizient durchgeführt werden kann. In der Praxis sind die Blattknoten üblicherweise keine Dezimalcodes von Zeichen nach dem ASCII-Standard, sondern die Hashwerte von zu schützenden, kritischen Daten. Zum Beispiel könnte sich der Wert 72 des ersten Blattknotens durch die Anwendung einer Hash-Funktion $\tilde{h}$ auf der folgenden fiktiven Darstellung einer Transaktion ergeben:

```
{
    "sender":       "0x42"
    "recipient":    "0x17"
    "timestamp":    "05.07.2023 00:34",
    "amount":       "19.041.992 ₵"
}
```

Das Dokument beschreibt eine Überweisung auf ein Konto 0x17 von 0x42 am 05.07.2023 über einen Betrag von 19.041.992 ₵. In einer Blockchain kann der Merkle-Baum dazu verwendet werden, die Transaktionen in einem Block zu verwalten.

Existenzprüfung

BEWEIS DER EXISTENZ EINES DOKUMENTS Es soll gezeigt werden, dass das obige Dokument in einem Datensatz enthalten ist, welcher mit dem Merkle-Baum in Abbildung 27 gesichert ist. Voraussetzung dafür ist, dass der Wert des Wurzelknotens (4) allgemein bekannt und akzeptiert ist. Des Weiteren wird der Hashwert des Dokuments benötigt (72).

Für den Beweis wird gezeigt, dass ein sogenannter *Pfad* von einem Blattknoten zur Wurzel des Merkle-Baums bekannt ist, mit welchem sich der Wert des akzeptierten Wurzelknotens verifizieren lässt. Im Beispiel würden für den Pfad die Werte 65, 2 und 5 bestimmt werden. Die korrespondierenden Knoten sind in Abbildung 27 rot markiert

und referenzieren den Geschwisterknoten des zu beweisenden Dokuments sowie die jeweiligen Geschwisterknoten auf den höheren Ebenen.

Für den Beweis lässt sich anhand dieses Pfades die Wurzel des Merkle-Baums neu berechnen und verifizieren:

$$h(72 + 65) = 7$$
$$h(7 + 2) = 9$$
$$h(9 + 5) = 4$$

Damit ist der Wert 4 des Wurzelknotens bestätigt.

Es ist zu beachten, dass dies kein Beweis im mathematischen Sinn ist, jedoch ist dieses Verfahren in der Praxis ausreichend, wenn sichere kryptografische Hash-Funktionen verwendet werden. Denn dann ist es aufgrund der Preimage Resistance-Eigenschaft praktisch nicht möglich, ein manipuliertes Dokument zu finden, welches den geforderten Hashwert (72) besitzt. Damit kann angenommen werden, dass das Dokument tatsächlich in dem betreffenden Datensatz existiert.

Integritätsprüfung

Integritätsüberprüfung eines Datensatzes Wenn ein Angreifer das oben gezeigte Dokument manipulieren will, indem der Betrag auf 21.041.961 ℂ erhöht wird, ändert sich damit auch der Hashwert des Dokuments, zum Beispiel auf den Wert 73. Wenn ein verdächtiges Dokument überprüft werden soll, kann wiederum der Merkle-Pfad bestimmt werden, um damit den Wert der Wurzel zu verifizieren. Mit dem rot markierten Merkle-Pfad in Abbildung 27 und einem Hashwert von 73 des manipulierten Dokuments würde für die Wurzel nun der Wert 5 berechnet werden:

$$h(h(h(73 + 65) + 2) + 5) = 5$$

Damit ist gezeigt, dass das Dokument verändert wurde. Bei Verwendung von sicheren kryptografischen Hash-Funktionen kann der Angreifer aufgrund der 2nd-Preimage Resistance-Eigenschaft praktisch auch kein zweites Dokument finden, welches denselben Hashwert (72) wie das Originaldokument aufweist.

Der Merkle-Baum wird im Blockchain-Protokoll Ethereum in einer modifizierten Variante als *Modified Merkle Patricia Tree* [190, Appendix D] an verschiedenen Stellen eingesetzt, darunter zur Verwaltung von Kontoständen [190, Kap. 4.1] oder zur Speicherung aller Transaktionen eines Blocks [190, Kap. 4.3].

4.2.4 *Proof of Work*

1993: Dwork und Naor

Cynthia Dwork und Moni Naor haben 1993 einen Ansatz veröffentlicht, der das übermäßige Senden von E-Mails beschränken sollte [40].

Die prinzipielle Idee ist, dass der Sender einer E-Mail für eine erfolgreiche Zustellung eine Aufgabe lösen muss. Für den praktikablen Einsatz muss diese Aufgabe so gewählt werden, dass sie einerseits mit vertretbarem Aufwand für den Sender verbunden ist und gleichzeitig das Ergebnis für den Empfänger beziehungsweise für den Server sehr einfach zu verifizieren ist. Nur bei einer erfolgreichen Verifikation des Ergebnisses wird die E-Mail schließlich zugestellt.

Eine mögliche Aufgabe, die von Dwork und Naor unter anderem vorgeschlagen ist, basiert auf der Quadratwurzelberechnung über einem Restklassenring. Dabei wird für ein Proof of Work-System eine Primzahl p gewählt. Außerdem sei

$$f_p : \mathbb{N} \to \mathbb{Z}_p, f_p(x) = \sqrt{x} \bmod p$$

Um einen Proof of Work zu generieren, muss für eine gegebene Primzahl p und eine Zahl x eine Zahl y gefunden werden, sodass $y^2 \equiv x \bmod p$. Dabei ist es für große Zahlen aufwendig, ein passendes y zu finden, wobei die Verifikation sehr effizient ist [130].

Beispiel: Proof of Work. Für das folgende Beispiel sei eine Primzahl $p = 23$ gegeben.

1. Ein Sender möchte eine E-Mail versenden und fragt dafür beim E-Mail-Server an.
2. Der E-Mail Server antwortet mit einer Zufallszahl $x = 2$ als Basis für den zu berechnenden Proof of Work.
3. Der Sender muss nun eine Zahl y finden, sodass $y^2 = 2 \bmod 23$.
4. Der Sender führt die Berechnung aus und erhält $y = 5$.
5. Der Sender verschickt die E-Mail, zusammen mit dem gefundenen Wert $y = 5$.
6. Der E-Mail-Server überprüft, ob $5^2 \equiv 2 \bmod 23$ und leitet die E-Mail an den Empfänger weiter.

1997: Beck
2002: Hashcash

Das Prinzip wurde 1997 von Adam Beck erweitert [10] und 2002 als *Hashcash* veröffentlicht [11]. Nach eigener Aussage wurde Hashcash unabhängig von der Arbeit von Dwork und Naor [40] zur Vermeidung von E-Mail-Spam entwickelt. Das Ziel von Hashcash ist es, den Zugriff auf frei verfügbare oder *un-metered* (deutsch: nicht gebührenpflichtige) Internetdienste wie E-Mail-Server zu beschränken. Dafür wurde ebenfalls ein Proof of Work-System entwickelt, welches ähnlich zu dem Verfahren von Dwork und Naor auf einer aufwendig zu berechnenden und leicht zu verifizierenden Aufgabe basiert. Bei Hashcash muss ein Client einem bestimmten Server, dessen Res-

sourcen genutzt werden sollen, gegenüber beweisen, eine aufwendige Berechnung durchgeführt zu haben. Dafür schlägt Hashcash vor, dass der Client partielle Hash-Kollisionen von kryptografischen Hash-Funktionen finden muss. Konkret bedeutet das, dass ein Client eine Eingabe x zu einer gegebenen kryptografischen Hash-Funktion $\mathcal{H}$ finden muss, sodass eine bestimmte Anzahl k an führenden Bits der Ausgabe der Hash-Funktion den Wert 0 annehmen. Kryptografische Hash-Funktionen haben die Eigenschaft, dass kein effizientes Verfahren existiert, zu einem gegebenen oder gewünschten Funktionswert $\mathcal{H}(x)$, eine valide Eingabe x zu bestimmen (Preimage Resistance). Es existiert also kein effizienteres Verfahren, als über einen Brute-Force-Algorithmus verschiedene Werte für x auszuprobieren.

Hashcash Implementierung

Quelltext 3 zeigt die Implementierung eines einfachen Proof of Work-Systems nach dem Vorbild von Hashcash. Der Funktion `proofOfWork` wird die Komplexität als Parameter übergeben, welcher die Anzahl der zu erzeugenden führenden Nullen festlegt (Zeile 1). Zu Beginn werden die folgenden Variablen initialisiert (Zeilen 2–5). Die Laufvariable `nonce`, eine Abbruchvariable `found` sowie die Hilfsvariable `target`, welche eine Zeichenkette mit der entsprechenden Anzahl an Nullen für eine spätere Überprüfung anlegt. In einer Schleife wird zuerst die Laufvariable inkrementiert (Zeile 8). Anschließend wird der aktuelle Zeitpunkt zusammen mit der Laufvariable gehasht (Zeilen 9–11). Falls das Ergebnis der Schwierigkeit entsprechend genügend führende Nullen hat, wird der Wert der Abbruchvariable umgekehrt und der Proof of Work wird zurückgegeben (Zeilen 13–18).

```
1  public String proofOfWork(int difficulty) {
2      int nonce = 0;
3      boolean found = false;
4      String target = "0".repeat( difficulty );
5      String hash = "";
6
7      do {
8          nonce++;
9          String timestamp = Instant.now().toString();
10         String proofString = timestamp + nonce;
11         hash = SHA256(proofString);
12
13         if(hash.substring(0, difficulty ).equals(target))
14             found = true;
15         }
16     while(!found);
17
18     return hash;
19 }
```

Quelltext 3: Einfache Proof of Work-Implementierung in Java

1999: Jakobsson
2008: Nakamoto

Der Begriff *Proof of Work* wurde im Jahr 1999 von Jakobsson und Jules eingeführt und formalisiert [79]. Knapp 10 Jahre später im Jahr

2008 wurde eine E-Mail versendet[6], in welcher die Arbeit an einem digitalen Geldsystem namens *Bitcoin* [133] veröffentlicht wurde. Der Absender dieser E-Mail ist Satoschi Nakamoto, ein Pseudonym, von dem bis heute nicht eindeutig geklärt ist, welche Person oder Personengruppe sich dahinter verbirgt. Das beschriebene Bitcoin-Geldsystem läuft in einem Peer-to-Peer-Netzwerk und benötigt keine vertrauenswürdige Instanzen, wie etwa Banken oder sonstige Geldinstitute. Insbesondere wird hervorgehoben, dass der aus Hashcash bekannte Proof of Work-Mechanismus für die Erzeugung von *Coins* (deutsch: Münzen) verwendet wird. Außerdem wird mit Bitcoin und dem integrierten Proof of Work-Mechanismus erstmals das *Double-spending*-Problem, also das mehrmalige Ausgeben derselben digitalen Münze gelöst [133], was unter anderem den Erfolg von Bitcoin besiegelte.

4.2.5 *Smart Contract*

Smart Contracts

Nick Szabo hatte im Jahre 1997 die Vision, vertragliche Konditionen automatisiert über Software oder Hardware als *Smart Contract* (deutsch: intelligenter, automatisierter Vertrag) umzusetzen [173]. In seiner Arbeit beschreibt er einen Verkaufsautomaten als eine einfache Form eines automatisierten Vertrags. Durch das Einwerfen einer Münze wird die Warenauswahl freigeschaltet, worauf der Kunde die Ware auswählt und anschließend erhält. Ein komplexeres Beispiel stammt aus dem interdisziplinären Bereich der Mobilität und dem Finanzwesen. Zum einen kann über eine Software die Funktionstüchtigkeit eines Autos beschränkt werden, wenn sich nicht der rechtmäßige Besitzer über ein *Challenge-Response*-Protokoll (deutsch: Aufforderung-Antwort-Verfahren) authentifiziert. Zum anderen kann, falls das Auto als Sicherheit bei einem Kreditverfahren eingesetzt wird, die Funktionalität bei einer ausbleibenden Rückzahlung eingeschränkt werden. In diesem Fall muss der Smart Contract aus Sicherheitsgründen überprüfen, ob sich das Fahrzeug aktuell auf einer Autobahn fortbewegt, bevor die Funktionen beschränkt werden.

Zum damaligen Zeitpunkt war die Digitalisierung jedoch noch nicht weit fortgeschritten und notwendige Hardwareressourcen nicht verfügbar, um die konzeptionellen Ideen umzusetzen. Dementsprechend gab es im Zuge dieser Pionierarbeit noch keine Implementierung. Erst in den 2010er Jahren wurde mit der Blockchain-Technologie eine Infrastruktur gefunden, mit der Smart Contracts automatisiert in dezentralen Netzwerken ausgeführt werden können [190].

6 https://www.mail-archive.com/cryptography@metzdowd.com/msg09959.html, besucht am 11.02.2024

4.3 FUNKTIONSWEISE EINER BLOCKCHAIN

Nachdem durch Kapitel 4.1 und Kapitel 4.2 die Blockchain als Black Box sowie die wichtigsten zugrundeliegenden Prinzipien vorgestellt sind, beleuchtet dieses Kapitel nun die Funktionsweise und intern ablaufenden Prozesse.

Abbildung 28 zeigt die wichtigsten Komponenten einer Blockchain-Anwendung, welche von einem Teilnehmer lokal auf dem System ausgeführt wird. Durch die Anwendung wird eine Verbindung zu anderen Knoten aufgebaut, welche die gleiche Anwendung ausführen, um damit ein Blockchain-Netzwerk zu bilden (dunkle Kreise und Kanten, rechts in Abbildung 28). Die folgenden Erläuterungen referenzieren weiterhin die Elemente in Abbildung 28 in Klammern.

Erstellen, Signieren und Senden von Transaktionen

SENDEN UND SIGNIEREN EINER TRANSAKTION Um Daten auf der Blockchain zu aktualisieren, zum Beispiel um eine Menge an Kryptowährung zu einem anderen Account zu transferieren, muss ein Nutzer über die Client-Anwendung eine Transaktion erstellen (lila Pentagon, rechts oben). Außerdem wählt er einen Account, mit dessen privaten Schlüssel die Transaktion signiert wird, sodass jeder andere Teilnehmer die Signatur über die Verifikation mit dem korrespondierenden öffentlichen Schlüssel des Sender-Accounts prüfen kann. Der Sender kann somit seine Autorenschaft der Transaktion nicht leugnen. Über das Protokoll wird diese Transaktion schließlich im Netzwerk an alle Knoten verteilt. Transaktionen können neben Überweisungen auch zum Bereitstellen (gelbe Pentagone) und Aufrufen (pinke Pentagone) von Smart Contracts genutzt werden.

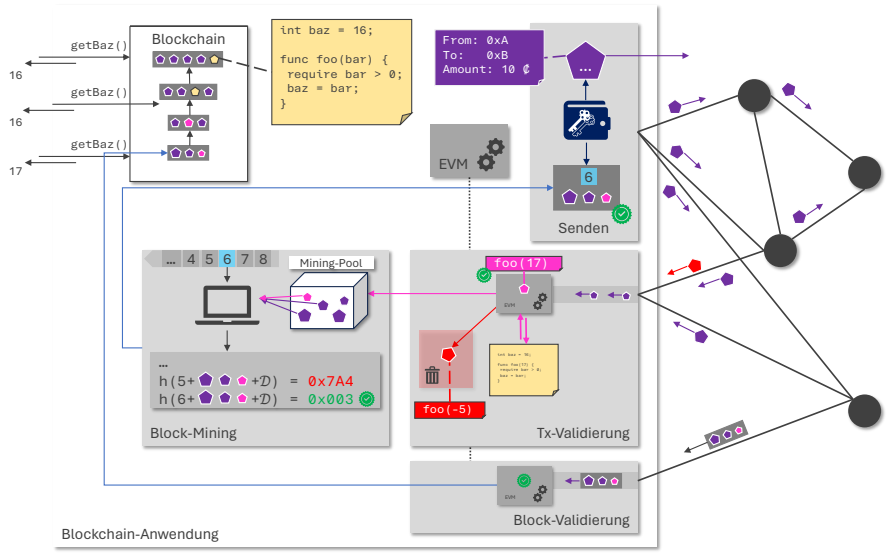

Abbildung 28: Intrinsische Prozesse von Blockchain-Systemen

syntaktische und semantische Korrektheit

VALIDIERUNG VON TRANSAKTIONEN Knoten überprüfen automatisch über die ausgeführte Client-Anwendung jede eingehende Transaktion auf syntaktische und semantische Korrektheit (graue Box Tx-Validierung, mittig). Dies umfasst, dass alle notwendigen Informationen in der erwarteten Struktur vorhanden sind oder dass der Sender-Account eine ausreichende Menge an Kryptowährung für die Überweisung kontrolliert. In dem Fall, dass die Transaktion eine Smart Contract-Funktion aufruft, zum Beispiel `foo(17)`, wird während der Validierung der entsprechende Smart Contract geladen und der Quelltext der Funktion ausgeführt, um zu überprüfen, ob die Ausführung erfolgreich beendet wird.

simulierte Smart Contract-Ausführung

Beispiel: Tx-Validierung. Im Beispiel in Abbildung 28 ist im ersten Block der Blockchain ein Smart Contract bereitgestellt (gelbes Rechteck, oben links). Dieser Smart Contract initialisiert eine Variable baz mit dem Wert 16 und stellt eine Funktion foo bereit. Diese Funktion erwartet einen Parameter bar und weist diesen Wert der Variable baz zu, sofern der Wert größer als 0 ist. Andernfalls wird durch das require-Schlüsselwort ein Fehler geworfen.

Während der Validierung wird der Quelltext testweise in einer abgekapselten Umgebung ausgeführt. Ist die Ausführung erfolgreich, wird die Transaktion in den sogenannten *Mining-Pool* aufgenommen, zum Beispiel die pinke Transaktion, welche die foo-Funktion mit dem Wert 17 aufruft. Eine weitere Transaktion (rot dargestellt) repräsentiert einen Funktionsaufruf von foo mit dem Parameter −5. Hier ist die require-Bedingung verletzt, sodass die Ausführung mit einem Fehler abgebrochen wird. Damit wird die Transaktion als nicht valide betrachtet und nicht weiter berücksichtigt.

Alle geprüften Transaktionen werden anschließend im Mining-Pool gesammelt und sind damit potenzielle Kandidaten für den nächsten Block.

Proof of Work

BLOCK-MINING Je nach konfigurierten Konsensverfahren darf potenziell jeder Knoten am Mining-Prozess teilnehmen, über welchen ein neuer Block gesucht wird. Beim Proof of Work-Verfahren werden dabei eine Menge an geprüften Transaktionen aus dem Mining-Pool ausgewählt und zusammen mit einer zufälligen Zahl, genannt *Nonce*, als Eingabedaten für eine Hash-Funktion verwendet. Das Ziel ist es, eine zufällige Zahl zu finden, sodass der resultierende Hashwert des Blocks einen bestimmten Schwellwert unterschreitet. Durch den Einsatz einer kryptografischen Hash-Funktion gibt es kein effizienteres Verfahren, als zufällige Werte für die Nonce auszuprobieren. Der Schwellwert, also die Wahrscheinlichkeit, mit der ein Block gefunden wird, wird automatisch über die im Netzwerk verfügbare Rechenleistung kalkuliert und stabilisiert eine Blockzeit von beispielsweise 10

Blockerstellung

Minuten im Bitcoin-Netzwerk.[7] Im Header wird dann unter anderem die Wurzel eines Merkle-Baums als Prüfsumme für alle Transaktionen im Block gespeichert. Dieser Hashwert wird dazu verwendet, andere Blöcke zu referenzieren. Genauer gesagt, im neu erstellten Block wird der Hashwert des vorherigen Blocks eingefügt, welcher die genauen Inhalte dessen repräsentiert. Dadurch wird eine Kette an Blöcken gebildet, welche bis zum ersten Block zurückreicht und Namensgeber für die Blockchain ist.

Beispiel: Block-Mining. In der grauen Box links unten in Abbildung 28 werden drei Transaktionen unter Berücksichtigung der im Protokoll spezifizierten Blockgröße aus dem Mining-Pool ausgewählt. Im Mining-Prozess dienen die Transaktionen zusammen mit einer Nonce $n \in \mathbb{N}$ und weiteren Daten $\mathcal{D}$, worin unter anderem eine Referenz auf den vorausgehenden Block enthalten ist, als Eingabe für eine Hash-Funktion h.

Es muss nun eine Nonce gefunden werden, sodass sich ein ausreichend kleiner Funktionswert von h ergibt, zum Beispiel ein Hashwert kleiner 10. Die Zahlen von 1 bis 5 brachten bisher keinen Erfolg, zum Beispiel ergab $h(5 + tx_1 + tx_2 + tx_3 + \mathcal{D})$ den Wert $7A4_{16}$ beziehungsweise 1956_{10}. Wenn hingegen der Wert 6 als Nonce verwendet wird, ergibt sich im Beispiel ein Hashwert von 3_{10}, wodurch ein valider Block gefunden ist, der unter anderem die Transaktion für den Funktionsaufruf `foo(17)` beinhaltet.

Verteilen des neuen Blocks

VALIDIERUNG VON BLÖCKEN Ein gefundener Block, der allen definierten Validierungsregeln entspricht, wird ähnlich den Transaktionen im gesamten Netzwerk verteilt. Nach dem Empfang des Blockvorschlags validieren die Knoten den Block und alle enthaltenen Transaktionen ähnlich wie in Box Tx-Validierung beschrieben. Bei erfolgreicher Validierung wird der Block der lokalen Kopie der Blockchain hinzugefügt (links oben in Abbildung 28). Dabei werden die darin enthaltenen Transaktionen interpretiert, das heißt, dass die Bilanzen der Accounts aktualisiert und die korrespondierenden Smart Contract-Funktionen mit finaler Auswirkung auf den Datenstand ausgeführt werden. Abbildung 28 visualisiert oben links, dass die Abfrage des baz-Wertes so lange den ursprünglichen Wert 16 zurückliefert bis eine Transaktion, welche den Wert über die foo-Funktion ändert, in einem validen Block aufgenommen ist. Die Transaktionen in diesem Block sind nun Teil des sogenannten Distributed Ledgers und werden aus den Mining-Pools der Knoten entfernt.

Ausführung der Transaktionen

Auswirkung auf den Distributed Ledger

Dadurch dass jeder ehrliche Knoten die Transaktionen und Blöcke nach den gleichen Regeln prüft, wird so ein global akzeptierter Stand der Blockchain verwaltet.

7 https://bitaps.com/, besucht am 27.11.2023

Ausführung von Smart Contract-Funktionen

ETHEREUM VIRTUAL MACHINE Die Ausführung der Funktionen von Smart Contracts erfolgt in der EVM (*Ethereum Virtual Machine*) [8, Kap. 13]. Dabei werden die Smart Contracts als Bytecode in die EVM geladen und die in der Transaktion codierten Maschinenbefehle interpretiert [8, S. 111–112]. Im Zuge der Validierung von Transaktionen ist ein deterministisches Verhalten der EVM essenziell, sodass unterschiedliche Knoten zu unterschiedlichen Ausführungszeitpunkten dennoch die gleichen Resultate während der Validierung erhalten und damit die gleichen Daten in die lokalen Datenstrukturen der Blockchain speichern. Dementsprechend sind manche Operationen wie das Aufrufen extrinsischer Schnittstellen zum Abrufen von Datenwerten in der EVM nicht umsetzbar, da der Determinismus dieser Dienste nicht vorausgesetzt werden kann. Um dennoch extrinsische Daten zu laden, können sogenannte *Events* (deutsch: Ereignisse) verwendet werden. Über Smart Contracts lassen sich Ereignisse über *Event emitter* auslösen, auf welche externe Dienste einen Event-Handler registrieren können, also eine Funktion, welche automatisch ausgeführt wird, wenn ein Ereignis ausgelöst wird. Ereignisse lassen sich über die Schlüsselwörter `event` und `emit` im Solidity-Quelltext zur Implementierung von Smart Contracts definieren und werden von der EVM ausgelöst. Aus technischer Sicht bedeutet dies, dass Transaktionen, welche eine ereignisauslösende Smart Contract-Funktion aufrufen, in ihrem transaktionseigenen Ereignisprotokoll (englisch: *transaction log*) einen entsprechenden Eintrag führen. Dieses Ereignisprotokoll wird anschließend in einem bestimmten Speicherbereich der Blockchain abgelegt, welcher von externen Diensten überwacht werden kann [8, S. 149–152].

deterministische EVM

Events

Auf Basis dieses Mechanismus können schließlich sogenannte *Oracles* definiert werden, um extrinsische Daten in das Blockchain-Universum zu laden. Oracles sind Smart Contracts, welche ein Ereignis aussenden, woraufhin ein externer Dienst eine Transaktion dieses Oracle-Smart Contracts aufruft, um mithilfe einer Transaktion zum Beispiel ein Datenfeld im Oracle zu aktualisieren. Auf diese Weise kann anderen Smart Contracts ein einheitlicher Zugriff auf diesen Datenwert gewährt werden.

Oracles

4.4 BEWERTUNG UND ZUSAMMENFASSUNG

Dezentralität und unbeschränkte, pseudonymisierte Nutzer

DEZENTRALISIERUNG Blockchain-Netzwerke wie Bitcoin oder Ethereum können für eine vollständig dezentrale Verwaltung konfiguriert werden und erlauben dabei eine theoretisch unbeschränkte und pseudonymisierte Teilnahme von Benutzern. Dies wird unter anderem durch die besondere Integritätsprüfung in der Blockchain-Datenstruktur, das Konsensverfahren Proof of Work und weiteren Eigenschaften, darunter digitale Signaturen, ermöglicht. Grundsätzlich existiert bei Blockchain-Systemen allerdings ein Trilemma aus Dezentrali-

Trilemma und Quadrilemma

sierung, Sicherheit, Skalierbarkeit [65, 126, 186], welches auch bereits um den Aspekt der Privatsphäre zu einem Quadrilemma erweitert ist [151]. Das bedeutet, dass bei hoher Dezentralisierung und Aufrechterhaltung der Sicherheit die Skalierbarkeit, gemessen an Transaktionen pro Sekunde, beeinträchtigt ist. Andererseits bedingt eine hohe Skalierbarkeit in einem sicheren Netzwerk eine (teilweise) Zentralisierung des Netzwerks. Dieser Aspekt wird unter dem Punkt ZENTRALISIERUNG weiter unten diskutiert.

Blockchain-Fork

PROBABILISTISCHE FINALITÄT UND LONGEST CHAIN RULE Aufgrund einer hohen Dezentralisierung des Netzwerks sind zeitweise Dateninkonsistenzen nicht ausgeschlossen und treten beispielsweise auf, wenn in dem globalen Netzwerk zwei verschiedene Knoten gleichzeitig einen validen Block erstellen, also eine geeignete Nonce für eine bestimmte Menge an Transaktionen finden. Dadurch wird das Netzwerk partitioniert und die unterschiedlichen Bereiche nehmen den einen oder den anderen Block als gültigen Nachfolger des aktuell gespeicherten Blocks an, je nachdem welcher Block zuerst an einem Knoten validiert wird. Man spricht dabei von sogenannten *Blockchain-Forks*.

Longest Chain Rule

In den Protokollen wird üblicherweise spezifiziert, wie eine derartige Netzwerkpartitionierung aufgelöst wird, nämlich indem jeder Knoten stets die längste Kette an Blöcken als die gültige Blockchain annimmt, zum Beispiel über die *Longest Chain Rule* (deutsch: Regel der längsten Kette) in Bitcoin [133] oder über den *Latest Message Driven Greedy Heaviest-Observed Sub-Tree (LDM GHOST)* (deutsch: Nachrichten getriebener gieriger schwerster beobachteter Teilbaum) in Ethereum [190, Kap. 11]. Das bedeutet, wenn zu einem bestimmten Zeitpunkt eine der Partitionen einen weiteren Nachfolger des lokal als gültig angenommen Blocks findet und an das gesamte Netzwerk propagiert, wird das gesamte Netzwerk diese längere, durch einen größeren Rechenaufwand gesicherte Blockchain als gültig betrachten und die eigene Version verwerfen.

probabilistische Finalität

Das Auflösen eines Blockchain-Forks kann schwerwiegende Auswirkungen haben. Die Blöcke in den unterschiedlichen Netzwerkpartitionen validieren unter Umständen unterschiedliche Transaktionen. Das bedeutet, dass durch das Auflösen des Blockchain-Forks und dem damit einhergehenden Ersetzen von gültigen Blöcken durch andere gültige Blöcke, bereits validierte Transaktionen nicht in der *neuen* Blockchain enthalten sind. Bereits als getätigt erachtete Überweisungen oder schon ausgeführte Funktionen in Smart Contracts und Datenänderungen werden dadurch revidiert. Man spricht bei diesem Phänomen auch von *probabilistischer Finalität* oder *probabilistischen Konsens* [174], da die Wahrscheinlichkeit für Transaktionen, final in der Blockchain persistiert zu sein, mit zunehmenden gefundenen

Blöcken steigt. Im Gegensatz dazu steht die *deterministische Finalität*, welche unten unter dem Punkt ZENTRALISIERUNG diskutiert wird.

monetärer Anreitz bei Proof of Work

ÖKONOMIE Im Protokoll der ersten Blockchain, Bitcoin, ist die Ökonomie der Kryptowährung essenziell für die Aufrechterhaltung des laufenden Betriebs. Knoten, welche sich am Proof of Work-Verfahren zur Validierung von Transaktionen beteiligen, werden für den zu leistenden Rechenaufwand in Form von Kryptowährung belohnt. Diese Belohnung dient als Anreiz (englisch: *Incentive*) dem Protokoll folgend zu handeln und überhaupt am Proof of Work-Verfahren teilzunehmen. Neben dieser Belohnung erhält der Blockersteller zudem noch die von den jeweiligen Sendern für die Validierung zu zahlenden Transaktionsgebühren.

Kosten von Smart Contracts

Im Ethereum-Protokoll spielen Kryptowährungen eine entscheidende Rolle bei der Definition und Ausführung von Smart Contracts. Zum einen skalieren die Gebühren von Transaktion, welche einen neuen Smart Contract bereitstellen, mit der Größe des Smart Contracts, um Netzwerkbehinderungen zum Beispiel durch unnötig große Smart Contracts zu verhindern. Zum anderen ist auch die Ausführung von Smart Contract-Funktionen gebührenpflichtig. In einer entsprechenden Transaktion wird vom Sender eine gewisse Menge an sogenanntem *Gas* spezifiziert, welches für die Abarbeitung der Maschineninstruktionen bei der Ausführung der Smart Contract-Funktion im Validierungsprozess zur Verfügung steht.[8] Außerdem spezifiziert der Sender einen variablen Gas-Preis, der die Menge an Kryptowährung pro Gas-Einheit angibt, die der Sender für die Validierung der Transaktion bereit ist zu bezahlen. Die entsprechende Menge an Kryptowährung erhält der Blockersteller. Über diesen Mechanismus werden Denial-of-Service-Angriffe verhindert, da die Validierung abgebrochen wird, wenn kein Gas für die Ausführung weiterer Instruktionen zur Verfügung steht.

Gas für Smart Contract-Ausführung

Gas-Preis

Skalierbarkeit durch Zentralisierung

ZENTRALISIERUNG Die Bitcoin-Blockchain überzeugt durch die Proof of Work- und Kryptowährung-getriebene Dezentralisierung. In Hinblick auf das oben erwähnte Trilemma können andere Blockchain-Protokolle wie Ethereum allerdings auch Einschränkungen im Grad der Dezentralisierung hinnehmen, um etwa die Skalierbarkeit, also die Menge an validierten Transaktionen pro Zeiteinheit, zu verbessern. Konkret bedeutet dies, dass anstelle der Proof of Work- oder Proof of Stake-Konsensverfahren, ein *Proof of Authority*-Konsensverfahren wie *Clique*[9] eingesetzt wird, oder dass die Synchronisierung über traditionellere BFT-Algorithmen erfolgt [44, 75]. Dabei wird lediglich eine auserwählte, vordefinierte Menge an Benutzerkonten dazu ermächtigt, neue Blöcke zu erstellen und vorzuschlagen. Durch

Proof of Authority und BFT

vordefinierte Menge an Miner

8 Bestimmte Maschineninstruktionen benötigen eine bestimmte Menge an Gas.

9 https://eips.ethereum.org/EIPS/eip-225, besucht am 13.02.2024

Kryptowährung ohne Gegenwert

Round Robin statt Proof of Work

deterministische Finalität

zwischenbetriebliche Prozessausführung

diese Zentralisierung der Macht auf wenige Knoten und dem Ersetzen von Proof of Work, kann auf Kryptowährungen, genauer gesagt deren realen Gegenwert in traditionelleren Währungen, verzichtet werden. Ein weiterer Aspekt betrifft die Finalität. Die Auswahl eines Kontos zur Blockerstellung kann anstatt über das zeitintensive Proof of Work-Verfahren effizient über die vordefinierte Menge an Benutzerkonten zum Beispiel in einem Rundlauf-Verfahren [75] erfolgen. Dadurch können Blockchain-Forks vermieden werden, sodass die Transaktionen statt mit einer probabilistischen Finalität mit einer deterministischen Finalität sofort als persistent in der Blockchain-Datenstruktur erachtet werden können.

Im Bereich der zwischenbetrieblichen Prozessausführung sind die Voraussetzungen, durch welche eine Proof of Work-getriebene Blockchain die Vorteile von pseudonymisierter und unbegrenzter Teilnahme gewinnbringend zur Geltung bringen kann, möglicherweise oftmals nicht erfüllt. Stattdessen sind sowohl die Geschäftspartner als auch die möglichen Akteure der Prozessausführung bekannt und zahlenmäßig begrenzt, sodass die zentralisierteren Blockchain-Netzwerke vielversprechend für den Einsatz in der zwischenbetrieblichen Prozessausführung sind [44, 166]. Inwieweit durch die im Gegensatz zu öffentlichen Proof of Work-getriebenen Blockchains eingeschränkte Dezentralisierung Sicherheitsrisiken mit sich bringt oder für welche Anwendungsfälle die ein oder andere Konfiguration besser geeignet ist, soll in dieser Arbeit nicht weiter diskutiert werden. Die Artefakte, welche in Kapitel 5 und Kapitel 6 vorgestellt werden, können unabhängig von der Konfiguration der Netzwerke eingesetzt werden, wobei die speziellen Eigenschaften, insbesondere die probabilistische Finalität beim Artefakt in Kapitel 6, bei den Konzepten gegebenenfalls berücksichtigt werden.

5

BLOCKCHAINS ALS UMGEBUNG FÜR DEZENTRALE PROZESSAUSFÜHRUNG

In Kapitel 4 werden die Grundlagen und Eigenschaften von Blockchain-Systemen vorgestellt. Insbesondere wegen der dezentralisierten Kontrolle über Daten und Anwendungen, ohne dabei einen bestimmten Teilnehmer zu bevorzugen, lassen sich Blockchain-Netzwerke gewinnbringend bei der zwischenbetrieblichen Prozessausführung einsetzen. Vor der Entwicklung von Blockchain-Systemen werden zwischenbetriebliche Prozesse unter anderem aufgrund einer fehlenden Koordinierungsinstanz beispielsweise über die Implementierung lokaler Prozesse umgesetzt, welche sich gegenseitig basierend auf einem Nachrichtenaustauschprotokoll benachrichtigten, um die Verantwortung entsprechend zu übergeben. Ein Blockchain-System kann über die dezentrale Kontrolle der Netzwerkteilnehmer die Rolle dieser Koordinierungsinstanz einnehmen, da die eingesetzten Synchronisierungsmechanismen im Zusammenspiel mit weiteren Faktoren eine global akzeptierte Datenbasis ermöglichen. Wenn die Kollaboration als Prozessmodell spezifiziert werden kann, bietet die Blockchain somit eine geeignete Plattform zur Ausführung eines zwischenbetrieblichen Prozessmodells.

Dieses Kapitel beschäftigt sich mit der Verwendung von Blockchain-Systemen für die dezentrale Ausführung von Prozessen und zeigt, wie die Prozessausführung auf Basis von Smart Contracts implementiert werden kann. Dies erfolgt im Rahmen eines ersten Zyklus der in Kapitel 1.3 beschriebenen DSR-Forschungsmethodik. Folglich wird zuerst in Kapitel 5.1 eine Wissensbasis für die darauffolgenden Schritte aufgebaut. Kapitel 5.2 untersucht die bisherigen Konzepte und Ansätze zur Blockchain-basierten Prozessausführung und identifiziert dabei bestimmte Schwächen und Probleme. Darauf basierend postuliert Kapitel 5.3 Anforderungen an ein zu entwickelndes Artefakt, um die beschriebenen Herausforderungen zu lösen. Das Artefakt wird in Kapitel 5.4 vorgestellt, während dessen Funktionsweise und Verwendung Gegenstand von Kapitel 5.5 sind. Zuletzt erfolgt in Kapitel 5.6 eine Evaluation des Artefakts, indem diskutiert wird, inwieweit die postulierten Anforderungen umgesetzt und die Herausforderungen gelöst werden.

Dieser erste von insgesamt zwei DSR-Zyklen wird in Kapitel 6 mit dem zweiten Durchlauf vervollständigt. Darin werden die in Kapitel 5.6 neu evaluierten beziehungsweise immer noch bestehenden Herausforderungen in eine neue Problembeschreibung aufgenommen.

5.1 KONTEXT UND WISSENSGRUNDLAGE

Der Forschungsbereich dieses DSR-Projekts ist im Bereich der zwischenbetrieblichen Prozessausführung angesiedelt. Insbesondere liegt der Fokus auf Architekturen zur Umsetzung der zwischenbetrieblichen Prozessausführung mithilfe der innovativen Blockchain-Technologie. Zu Beginn wird eine *Knowledge Base* (deutsch: Wissensgrundlage) für dieses Thema erarbeitet, um eine Basis für die Identifikation von Schwachstellen aktueller Ansätze zu schaffen.

5.1.1 *Einordnung der Blockchain-basierten Prozessausführung*

Blockchain-basierte Prozessausführung versus nachrichtenbasierte Kollaboration

Vor der Blockchain-basierten Prozessausführung wurden zwischenbetriebliche Prozesse im Vergleich zu innerbetrieblichen Prozessen konzeptionell unterschiedlich betrachtet. Der Grund dafür liegt in den damals zur Verfügung stehenden Infrastrukturen: während innerhalb von Organisationsgrenzen eine zentral organisierte Client-Server-Infrastruktur verwaltet wird, kann eine derartige Infrastruktur für zwischenbetriebliche Prozesse nicht vorausgesetzt werden. Dementsprechend war die Bereitstellung eines Prozessausführungssystems für die Ausführung zwischenbetrieblicher Prozesse ohne Einbindung einer verwaltenden Autorität kompliziert. Aufgrund von Kosten oder Sicherheitsbedenken wurden zwischenbetriebliche Prozesse daher eher nachrichtenbasiert umgesetzt, mit der Folge, dass die gängigen Funktionalitäten von WFMSs nicht genutzt werden konnten (Kapitel 3).

Mit der Implementierung der Blockchain-Technologie in die Prozessausführung ist jedoch eine nicht manipulierbare Infrastruktur gefunden. Diese Infrastruktur bevorzugt oder benachteiligt keinen Teilnehmer und bietet eine global akzeptierte Datengrundlage für den aktuellen Prozessstatus und damit für die Prozessausführung.

5.1.2 *Blockchain-basierte Prozessausführung: kompilierender Ansatz*

Das Ethereum-Protokoll beschreibt das erste Blockchain-System, welches mit Smart Contracts neben einfachen Transaktionen auch kleine Programme ausführen kann und sich somit für eine fortgeschrittene Implementierung der Prozessausführung eignet.

kompilierender Ansatz nach Weber et al.

Eine der ersten wegweisenden Publikationen bei der Erforschung der Rolle der Blockchain im Bereich des Prozessmanagements wurde im Jahr 2016 veröffentlicht [185], woraus sich ein sehr dynamisches Forschungsgebiet entwickelte. Vor allem ab dem Jahr 2018 wurden

einige verschiedene Ansätze veröffentlicht. Eine detaillierte Analyse der verwandten Arbeiten erfolgt im Kapitel 8. Dieser Abschnitt konzentriert sich auf einen konkreten Ansatz, der als Grundlage für diesen DSR-Prozess dient, insbesondere für die Phase der Problembeschreibung. Dieser sogenannte *kompilierende Ansatz* wird in verschiedenen wissenschaftlichen Publikationen verfolgt, welche alle auf der ursprünglichen Idee von Weber et al. [185] aufbauen. Dieses Kapitel beschreibt die konzeptionelle Idee unabhängig von konkreten Implementierungen oder Erweiterungen.

Übersetzung eines Prozessmodells in einen Smart Contract

Prinzipiell wird beim kompilierenden Ansatz ein Prozessmodell in eine Blockchain-verständliche Form übersetzt, zum Beispiel in einen Smart Contract. Das resultierende Artefakt wird anschließend unterstützend für die Umsetzung der Kollaboration verwendet. Die Funktionsweise des kompilierenden Ansatzes lässt sich in drei beziehungsweise vier Phasen einteilen, welche im Folgenden vorgestellt werden. Die Erläuterungen basieren, ähnlich zur Arbeit von García-Bañuelos et al. [53], auf der Verwendung von BPMN-Prozessdiagrammen als Modellierungssprache sowie Ethereum als Implementierungs- und Ausführungsinfrastruktur.

Vorbereitung und Prozessmodellierung

PHASE 0 In einer Vorbereitungsphase wird sowohl das zwischenbetriebliche Prozessmodell als auch gegebenenfalls das dazugehörige Organisationsmodell erstellt. Speziell müssen hier auch Entscheidungen bezüglich der Konfiguration der Blockchain wie die Wahl eines Konsensverfahrens getroffen werden. Weiterhin müssen die Benutzerkonten für die möglichen Prozessteilnehmer oder für die Organisationen erstellt werden. In dieser Hinsicht muss auch unter anderem die Netzwerkinfrastruktur geklärt werden, unter anderem, ob jeder der möglichen Prozessteilnehmer einen Blockchain-Knoten verwaltet oder ob nur je teilnehmender Organisation ein Blockchain-Knoten zur Verfügung gestellt werden muss. Diese eher praxisrelevanten und anwendungsspezifischen vorbereitenden Maßnahmen sollen hier jedoch nicht weiter diskutiert werden.

Übersetzung des Prozessmodells

PHASE 1 In der ersten Phase wird das BPMN-Prozessmodell in eine Form übersetzt, welche von der Ethereum-Blockchain verarbeitet werden kann. Dazu wird ein Smart Contract verwendet, welcher in der Programmiersprache Solidity implementiert ist.

Der kompilierende Ansatz nutzt dabei ein *Transpiler*-Modul (deutsch: Quer-Übersetzer), welches aus dem Prozessmodell einen Smart Contract erstellt und die im Prozessmodell definierte Semantik direkt in die Solidity-Sprache übersetzt. Dabei werden unter anderem für jede Aktivität im Prozessmodell zwei Artefakte im Smart Contract generiert, nämlich eine Variable und eine Funktion. Die Variable zeigt an, ob der jeweilige Prozessschritt bereits erledigt ist, woraus der aktuelle Prozessfortschritt abgeleitet werden kann. Im Beispiel in Abbil-

dung 29 wird für das Prozessmodell der generierte Smart Contract in Pseudocode gezeigt. In den ersten zwei Zeilen ist für jede Aktivität des Prozessmodells je eine boolesche Variable mit dem Wert `false` initialisiert. Des Weiteren wird eine Funktion `A()` bereitgestellt, welche den Wert der entsprechenden Variable `A` auf `true` setzt, um anzuzeigen, dass die korrespondierende Aktivität abgeschlossen ist. Im Anschluss kann die Funktion `B` aufgerufen werden, welche zuerst prüft, ob die Aktivität A bereits ausgeführt ist. Ist dies der Fall, wird auch der Wert der Variablen `B` auf `true` gesetzt.

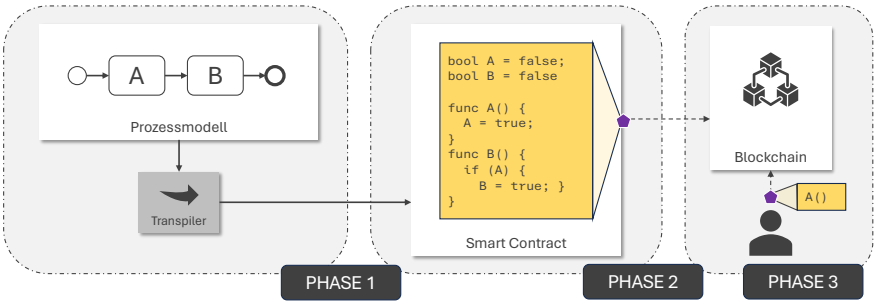

Abbildung 29: Kompilierender Ansatz zur Blockchain-basierten Prozessausführung

Die Implementierung der Variable kann entweder über einen booleschen Datentyp erfolgen, was hinsichtlich des Speicherplatzbedarfs eine effiziente Lösung ist. Die Verwendung komplexerer Datentypen erlaubt hingegen die Abspeicherung von zusätzlichen Informationen, etwa ein Zeitstempel oder die Ressource, welche dem Prozessschritt bei der Ausführung zugeordnet war.

Bereitstellung der Smart Contracts

PHASE 2 In der zweiten Phase werden die kompilierten Artefakte auf der Blockchain bereitgestellt. Dabei wird der generierte Smart Contract über eine Transaktion in der Blockchain, also in allen lokalen Kopien der Blockchain-Datenstruktur, persistiert, sodass jeder Teilnehmer auf den Smart Contract zugreifen und dessen Funktionen ausführen kann.

Ausführung über Smart Contract-Funktionen

PHASE 3 Die dritte Phase beschreibt die tatsächliche Ausführung. Hierbei werden bei diesem Ansatz die im Smart Contract generierten Funktionen aufgerufen, um dadurch die Variablen und den Status des Prozessfortschritts zu aktualisieren. Wenn ein Prozessschritt als erledigt gekennzeichnet werden soll, um damit den Prozessstatus zu aktualisieren, wird die Funktion aufgerufen, welche die entsprechende Variable aktualisiert. Um die Prozesskonformität zu gewährleisten, prüfen die Funktionen jeweils, ob die Variablen bezüglich der vorherigen Prozessschritte bereits den erfolgreichen Abschluss der erforderlichen Aktivität anzeigen.

5.2 PROBLEMSTELLUNG

Mit dem kompilierenden Ansatz ist gezeigt, dass die Ethereum-Blockchain gewinnbringend für die Umsetzung von zwischenbetrieblichen Geschäftsprozessen eingesetzt werden kann [53, 185]. Allerdings lassen sich auch Schwächen dieses Ansatzes identifizieren, welche im Folgenden diskutiert werden.

5.2.1 *Kompilierung und Interpretierung*

Kompilierung und Interpretierung in der Programmierung

Ähnlich zur Abgrenzung des kompilierenden Ansatzes von der traditionellen Sichtweise der Prozessausführung (Kapitel 2.3) kann man in der Softwareentwicklungsdisziplin zwischen kompilierenden und interpretierenden Programmiersprachen unterscheiden [155, S. 17]. Danach erstellt bei kompilierenden Programmiersprachen ein *Compiler* (deutsch: Übersetzer) Anwendungen in Form von fixen Anweisungen für ein bestimmtes Zielsystem, wodurch etwa Optimierungsschritte die Performance bei der späteren Programmausführung vorteilhaft beeinflussen können. Im Gegenentwurf der interpretierten Programme ist ein *Interpreter* zur Laufzeit des Programms aktiv und trifft Entscheidungen dynamisch basierend auf den Eingabedaten. Die Verlagerung der Entscheidungsfindung zur Ausführungszeit wird durch den Begriff *late binding* beschrieben, wodurch der Interpreter die Anweisungen des Quelltextes für jede Programmausführung neu analysiert. Statt der Performanceoptimierung wird bei der Interpretierung eine größere Flexibilität erreicht [155, S. 117].

Diese Prinzipien können auf die Prozessausführung angewendet werden. Statt vom Quelltext eines Programms ausgehend, welcher entweder in Maschinencode übersetzt oder von einem Interpreter zur Laufzeit analysiert wird, betrachtet man bei der Prozessausführung das Prozessmodell als Ausgangspunkt.

Kompilierung und Prozessausführung

Auf der einen Seite wird beim kompilierenden Ansatz basierend auf dem Prozessmodell ein Artefakt erstellt oder *kompiliert*, worin die Logik von dem spezifischen Ausgangsprozessmodell und gegebenenfalls die Informationen aus dem korrespondierenden Organisationsmodell kodiert ist. Dabei lassen sich zwei Artefakttypen unterscheiden. Das aus dem Prozessmodell generierte Artefakt ist entweder vom Typ eines zu interpretierenden Quelltextes wie JavaScript oder Quelltext einer Programmiersprache, welcher für sich wiederum in ausführbaren Maschinencode kompiliert werden muss, zum Beispiel Solidity-Quelltext zur Kompilierung für die EVM. In dieser Hinsicht ist die Bezeichnung *modellgetriebener Ansatz* präziser im Vergleich zu *kompilierender Ansatz*, da das generierte Artefakt entweder interpretiert oder weiter kompiliert werden kann. In dieser Arbeit wird weiterhin der in der Literatur gängige Begriff des kompilierenden Ansatzes verwendet. Auf der anderen Seite führt Kapitel 2.3 die

kompilierender oder modellgetriebener Ansatz

Interpretierung und Prozessausführung

Prozessausführung in einer Art und Weise ein, welche eher dem interpretierenden Ansatz in der Softwareentwicklung entspricht. Denn danach wird das Prozessmodell, gegebenenfalls zusammen mit dem Organisationsmodell, aktiv zur Laufzeit interpretiert und die Entscheidungen, zum Beispiel welche Aufgabe der Arbeitsliste welches Mitarbeiters hinzugefügt wird, muss dynamisch getroffen werden. Diese Vorgehensweise ist bei der Prozessausführung essenziell, da die verarbeiteten Datenwerte oder Verfügbarkeiten von Personen stetigen Änderungen unterliegen und zu der Kompilierzeit noch nicht feststehen. Darüber hinaus soll das Prozessmodell unter Umständen selbst zur Ausführungszeit angepasst werden können. Diese Flexibilität, Prozesse für bestimmte Anwendungsfälle zu konfigurieren, Aktivitäten und Kontrollflüsse zur Laufzeit zu ändern oder auf externe Ereignisse reagieren zu können, ist eine obligatorische Anforderung bei der Prozessausführung [144, Kap. 3.2].

Flexibilität

Zusammengefasst erlaubt es ein interpretierender Ansatz, die Anforderungen aus der Geschäftsprozessmanagementdomäne hinsichtlich der Flexibilitätsanforderungen besser umzusetzen.

5.2.2 *Kompilierung und Blockchain-basierte Ausführung*

Nachdem Kapitel 5.2.1 die beiden Paradigmen des kompilierenden Ansatzes und interpretierenden Ansatzes im Allgemeinen diskutiert, wird im Folgenden auf die möglichen Nachteile eingegangen, wenn ein kompilierender Ansatz auf einer Blockchain-Infrastruktur zur Anwendung kommt.

SKALIERBARKEIT Die Architektur skaliert schlecht mit der Größe der Prozessmodelle. Beim kompilierenden Ansatz werden für jeden Prozessschritt zwei Artefakte, je eine Variable und eine Funktion, generiert. Bei sehr großen Prozessmodellen werden dementsprechend viele Variablen und Funktionen im Smart Contract generiert. Dies hat folgende Auswirkungen.

Skalierbarkeit hinsichtlich Wartbarkeit

Mit einer steigenden Menge an Quelltext sinkt bei einer Software prinzipiell die Wartbarkeit, da der Aufwand zum Lesen, Ändern und Testen steigt [181, S. 131]. Allerdings findet der Aspekt in diesem Fall nicht unbedingt Anwendung, da beim kompilierenden Ansatz der generierte Quelltext per se nicht mehr angepasst wird. Sobald der Smart Contract auf der Blockchain bereitgestellt ist, kann dieser aufgrund der Datenstruktur darüber hinaus nicht mehr verändert werden. Bei veränderten Anforderungen oder der Integration weiterer Prozessperspektiven muss anstatt des generierten Quelltextes das Transpiler-Modul angepasst werden. Auf diese Herausforderung wird unten beim Punkt *Änderungsverwaltung* genauer eingegangen.

Skalierbarkeit hinsichtlich Kosten

Neben der Wartbarkeit spielen hinsichtlich der Skalierbarkeit im Blockchain-Universum auch die Kosten eine Rolle, denn die Bereit-

stellungskosten für einen Smart Contract sind von der Bytegröße des Smart Contract-Quelltexts abhängig [189, Formel (114)]. Es ist somit ratsam, den Quelltext für einen Smart Contract hinsichtlich der Größe zu optimieren [53]. Da beim kompilierenden Ansatz für viele Prozessschritte bei großen Prozessmodellen viele Variablen und Funktionen generiert werden, steigen insgesamt die Kosten für die Prozessausführung mit größeren Prozessmodellen. Auch die Relevanz von diesem Aspekt soll kurz diskutiert werden. Zum Zeitpunkt des Schreibens dieser Dissertation kostet eine Prozessausführung mit den aktuellen Ansätzen auf der öffentlichen Ethereum-Blockchain circa 100 Euro.[1] Ob Kosten in dieser Größenordnung für die Blockchain-basierte Koordination bei einer zwischenbetrieblichen Kollaboration gerechtfertigt sind, soll in dieser Arbeit nicht weiter diskutiert werden. Allerdings bieten private Blockchain-Netzwerke, welche beispielsweise mit einem Proof of Authority-Konsensverfahren konfiguriert sind und worin die Kryptowährungen keinen realen Gegenwert haben, eine Alternative für den Einsatz zur zwischenbetrieblichen Prozessausführung. Damit relativieren sich die Auswirkungen der Menge des generierten Quelltextes hinsichtlich der Kosten.

eingeschränkter Funktionsumfang bei existierenden Ansätzen

ERWEITERBARKEIT Eine Erweiterung der Architektur, welche den Funktionsumfang bestimmt, ist möglich, aber umständlich. In den ersten Ansätzen ist nur ein sehr eingeschränkter Funktionsumfang der Prozessausführung unterstützt. Unter anderem waren eine fortschrittliche Umsetzung der organisationalen und informationsorientierten Prozessperspektive in den Arbeiten von Weber et al. [185] oder García-Bañuelos et al. [53] nicht integriert. Aufgrund der Turing-Vollständigkeit bietet die EVM eine sehr mächtige Ausführungsumgebung, sodass fast beliebige Funktionalitäten für die Prozessausführung implementiert werden können [189, S. 13]. Wenn etwa die organisationale Perspektive in einem kompilierenden Ansatz unterstützt werden soll, erfordert dies jedoch eine aufwendige Integration im Transpiler-Modul. Insbesondere ist eine Repräsentation des organisationalen Modells im Solidity-Quelltext erforderlich, was wiederum auch negative Auswirkungen auf die Flexibilität und Skalierbarkeit hat.

aufwändige Transpiler-Anpassungen

Änderungen im Prozess- oder Organisationsmodell

ÄNDERUNGSVERWALTUNG Änderungen im Prozessmodell oder im Organisationsmodell erfordern beim kompilierenden Ansatz eine Neugenerierung des gesamten Smart Contract-Quelltexts. Dies hat zur Folge, dass sowohl bei Änderungen in der Organisationsstruktur, zum Beispiel aufgrund einer Beförderung, neuen Mitarbeitern oder Änderungen im Prozess selbst das Artefakt neu generiert werden muss.

1 Laut https://etherscan.io/gastracker, besucht am 30.11.2023, liegt der durchschnittliche Gas-Preis bei $46 \cdot 10^{-9}$ ETH und der ETH Preis bei circa 2.000 EUR, woraus sich bei einem Gas-Verbrauch von 1.201.358 (Wert aus [111]) ein Preis von $1 \cdot 10^{6} \cdot 2.000 \cdot 46 \cdot 10^{-9} = 92$ EUR ergibt.

Konsequenterweise erfordert dies auch, dass der generierte Smart Contract neu auf der Blockchain bereitgestellt werden muss, was wiederum gegebenenfalls mit Kosten verbunden ist.

Zusammenfassend zeigen sich beim kompilierenden Ansatz Defizite hinsichtlich der Skalierbarkeit, Erweiterbarkeit, Wartung und Flexibilität. Bei der Implementierung innerhalb eines Blockchain-Protokolls können diese Nachteile potenziell noch aufgrund des Kostenfaktors verstärkt werden. In den folgenden Kapiteln werden diese Problemstellungen adressiert und ein Artefakt für einen *interpretierenden Ansatz* im Rahmen der Blockchain-basierten Prozessausführung entwickelt.

5.3 ANFORDERUNGSDEFINITION

Basierend auf den Erkenntnissen der Grundlagen der zwischenbetrieblichen Prozessausführung aus Kapitel 3, der aufgebauten Wissensgrundlage in Kapitel 5.1 und den identifizierten Schwachstellen existierender Ansätze in Kapitel 5.2, werden im Folgenden die Anforderungen an das zu entwickelnde Artefakt postuliert. Die Anforderungen A1 und A2 leiten sich dabei aus den verwandten Arbeiten ab [53, 110, 185]. Die Anforderungen A3 und A4 beschreiben die konzeptionelle Weiterentwicklung der existierenden Ansätze in Richtung Interpretation eines Prozessmodells. Anforderung A5 ergibt sich schließlich durch die Umsetzungen der Anforderungen A3 und A4.

zwischenbetrieblicher Prozess als BPMN-Prozessdiagramm

A1: VERWENDUNG VON BPMN PROZESSDIAGRAMMEN BPMN gilt als de-facto Standard im Geschäftsmanagementbereich und stellt Prozessdiagramme zum Zwecke der Modellierung innerbetrieblicher, zentral orchestrierter Prozesse zur Verfügung, während Choreografien und Kollaborationsdiagramme für zwischenbetriebliche Prozesse verwendet werden. Dieses DSR-Projekt bricht mit dieser Konvention und stellt die Verwendung von BPMN-Prozessdiagrammen für die Modellierung zwischenbetrieblicher Prozesse als Anforderung. Der Grund dafür ist, dass BPMN die zwischenbetriebliche Prozessausführung als nachrichtenbasierte Kollaboration annimmt, während das zu entwickelnde Artefakt eine prozessbasierte Kollaboration erlauben soll. Dadurch können die verschiedenen Prozessperspektiven im Modell abgebildet werden und die Vorteile der WFMS-basierten Prozessausführung genutzt werden. Die Umsetzung erfolgt durch den vom Blockchain-System zur Verfügung gestellten und von allen Teilnehmern akzeptierten globalen Datenbestand, wodurch die orchestrierte Ausführung eines Prozessmodells ermöglicht wird. Das Zielartefakt übernimmt somit die Rolle eines WFMS für die Prozessausführung, so wie in Kapitel 2 vorgestellt.

Ethereum als Ausführungsplattform

A2: VERWENDUNG DER ETHEREUM-BLOCKCHAIN Das Ziel des DSR-Projekts ist es nicht, verschiedene Blockchain-Protokolle nach deren Eignung für die Prozessausführung zu evaluieren. Stattdessen soll auf das Ethereum-Protokoll zurückgegriffen werden, welches bereits in den Ansätzen verwendet wird, auf welchen die Problemidentifikation in Kapitel 5.2 basiert. Dabei soll die Flexibilität, fast beliebige Validierungsregeln über Smart Contracts zu definieren, die Entwicklung eines Interpreters für Prozessmodelle ermöglichen.

Interpretation des Prozessmodells

A3: INTERPRETIERENDE AUSFÜHRUNG NACH WFMS-VORBILD Im Gegensatz zum kompilierenden Ansatz von existierenden Ansätzen [53, 185], soll die Ausführung nach dem Vorbild der zentralisierten WFMS-basierten Prozessausführung nach Kapitel 2 konzipiert sein. Das bedeutet, dass statt der Kompilierung eines Prozessmodells in ausführbaren Quelltext, eine Repräsentation des Modells zu Ausführungszeit aktiv interpretiert werden soll. Insbesondere sollen dadurch die grundlegendsten Funktionalitäten von WFMSs aus Kapitel 2.4 erhalten bleiben, darunter die Überwachung des globalen Prozessstatus oder personalisierte Arbeitslisten.

Erweiterbarkeit

A4: ERWEITERBARKEIT FÜR WEITERE PROZESSPERSPEKTIVEN Kapitel 2.5 stellt die wichtigsten Prozessperspektiven vor und verdeutlicht deren Mehrwert bei der Prozessausführung. Durch die Konzeption als interpretierender Ansatz soll das Artefakt leicht hinsichtlich weiteren Perspektiven, etwa durch die organisationale oder durch die informationsorientierte Perspektive, erweiterbar sein.

Skalierbarkeit

A5: SKALIERBARKEIT FÜR GROSSE PROZESSMODELLE Die letzte Anforderung bezieht sich auf das Problem der Skalierbarkeit der kompilierenden Ansätze. Die Skalierbarkeit soll dadurch erreicht werden, dass die Bytegröße des Quelltextes des entwickelten Smart Contracts auch für sehr große Prozessmodelle nicht sehr stark ansteigt.

5.4 ETHEREUM-BASIERTE PROZESSAUSFÜHRUNG: INTERPRETIERENDER ANSATZ

DSR: Design & Development

Das Ziel eines DSR-Projekts ist die Definition von Anforderungen basierend auf einer Problemstellung und daraufhin die Entwicklung eines Artefakts, welches die definierten Anforderungen erfüllt. Das Artefakt wird am Ende gegen diese Anforderungen evaluiert, um gegebenenfalls offen gebliebene oder weitere Problemstellungen für einen weiteren DSR-Zyklus zu identifizieren. Das Artefakt wird im DSR-Schritt *Design & Development* entwickelt, worauf sich dieses Kapitel fokussiert. Im Folgenden wird das Konzept und die Architektur des Artefakts beschrieben.

5.4.1 *Entwurf und Konzeption des Artefakts*

Artefakt als Smart Contract

Nach Anforderung A2 wird das Artefakt gezielt für die Ethereum-Blockchain entwickelt. Das Artefakt wird somit von der Form eines oder mehrerer Smart Contracts sein, welche die interpretierende Prozessausführung nach Anforderung A3 umsetzen. Abbildung 30 gibt einen Gesamtüberblick über das Konzept.

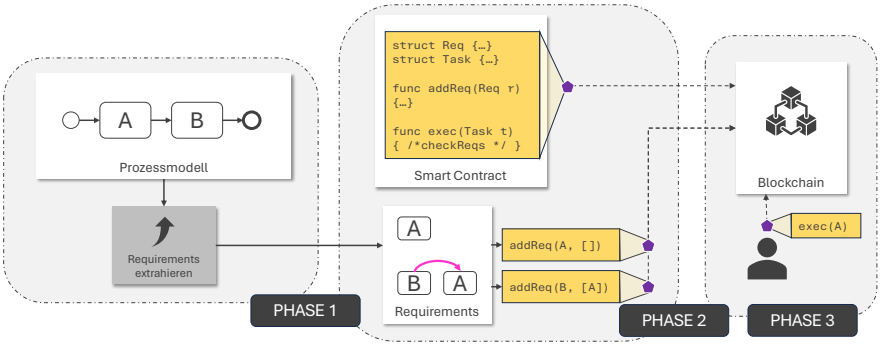

Abbildung 30: Interpretierender Ansatz zur Blockchain-basierten Prozessausführung

Im Zentrum des Konzepts steht ein Smart Contract, der für die Interpretation von BPMN-Prozessmodellen (Anforderung A1) verantwortlich ist. Der Smart Contract stellt die rudimentären Funktionalitäten eines WFMS bereit. Durch die Konzeption als Interpreter ist der Quelltext des Smart Contracts anfangs für alle Prozessmodelle gleich. Ähnlich einem WFMS wird zuerst der allgemeine Smart Contract *installiert*, also auf der Blockchain bereitgestellt, um anschließend das Prozessmodell darüber zu verwalten und zu instanziieren. Ähnlich wie beim kompilierenden Ansatz (Kapitel 5.1.2) erfolgt die Umsetzung in drei Phasen (Abbildung 30).

Übersetzung: Extraktion von Requirements

PHASE 1 Phase 1 ist dafür zuständig, das BPMN-Modell in eine Form zu übersetzen, welche von der Ethereum-Blockchain als Zielarchitektur verarbeitet werden kann. Statt wie beim kompilierenden Ansatz direkt einen spezifischen Smart Contract für das jeweilige Prozessmodell zu generieren, werden bei diesem Ansatz sogenannte *Requirements* (deutsch: Voraussetzungen) aus dem Prozessmodell extrahiert, welche die Semantik des Prozessmodells widerspiegeln. Diese Liste an Requirements ist die implementierungsspezifische Umsetzung der Prozessmodelllogik. In Analogie verwendet das Camunda WFMS etwa sogenannte *Executions* oder *ActivityInstances*, um die Semantik von BPMN umzusetzen.[2] Dabei wird zum Beispiel für jede Aktivität, welche einem parallelen Gatter folgt, jeweils eine *Execution* erstellt, welche dann für die Ausführung bereit ist. In Abbildung 30

2 https://docs.camunda.org/manual/7.20/user-guide/process-engine/process-engine-concepts/#executions, besucht am 03.12.2023

wird aus dem Prozessmodell ein Requirement B↷A, abgeleitet, das heißt, dass die Aktivität A abgeschlossen sein muss, bevor die Aktivität B starten kann. Zur Ausführung der Aktivität A hingegen müssen keine bestimmten Voraussetzungen erfüllt sein.

PHASE 2 In Phase 2 werden die Artefakte auf der Blockchain zur Verfügung gestellt. Statt einen für das jeweilige Prozessmodell spezifischen Smart Contract auf die Blockchain zu schreiben, wird beim interpretierenden Ansatz zuerst ein generischer Smart Contract bereitgestellt, der für alle Prozessmodelle gleich ist. Der Smart Contract stellt einerseits die WFMS-Funktionalitäten für die Prozessausführung wie die Methode `exec(Task t)` zur Aktualisierung des Lebenszyklusstatus einer Aktivität zur Verfügung. Andererseits bietet der Smart Contract mit der Funktion `addReq(Task t, Req[] reqs)` die Möglichkeit, die extrahierten Requirements hinzuzufügen, welche dann im Smart Contract gespeichert werden. Sobald der Smart Contract also bereitgestellt ist, werden aus den Requirements Blockchain-Transaktionen erstellt und der Reihe nach als Parameter der `addReq`-Funktion übergeben. Dadurch wird die Prozessmodelllogik in die Struktur des generischen Smart Contracts *hineingegossen*.

Bereitstellung eines generischen Smart Contracts

Bereitstellung des Prozessmodells als Requirements

PHASE 3 In Phase 3 erfolgt schließlich die Ausführung, welche wiederum über Transaktionen abgewickelt wird. Anstatt einer spezifischen Funktion für speziell die auszuführende Aktivität beim kompilierenden Ansatz (zum Beispiel `A()`), rufen die Benutzer nun die Funktion `exec(Task t)` auf. Die Aktivität, deren Status auf erledigt gesetzt werden soll, wird dabei als Parameter übergeben.

Ausführung über generische Funktion

5.4.2 *Entwicklung*

Dieses Kapitel stellt eine Implementierung des im vorherigen Kapitel präsentierten Konzepts vor. Das Konzept des interpretierenden Ansatzes soll im Gegensatz zu den bisher existierenden, kompilierenden Ansätzen eine WFMS-ähnliche Interpretation von BPMN-Prozessmodellen auf der Ethereum-Blockchain ermöglichen. Damit sollen auch die Vorteile eines WFMS (Kapitel 2) genutzt werden können, was in Kapitel 5.6 zu evaluieren ist. Für die Implementierung sind folgende Fragestellungen, welche oben teilweise bereits auf konzeptioneller Ebene betrachtet sind, jetzt im Detail zu beantworten:

- Wie wird die Logik des Prozessmodells im Smart Contract abgebildet? (Requirements-Konzept in Kapitel 5.4.2.1)
- Wie wird das Prozessmodell im Smart Contract bereitgestellt? (Transformation von BPMN in EVM-verständliche Form in Kapitel 5.4.2.2)

- Wie speichert und aktualisiert der Smart Contract den Prozessstatus? (Smart Contract in Kapitel 5.4.2.3)

5.4.2.1 *Abbildung der Semantik über Requirements*

Requirements mit `require`

Die Sprache Solidity definiert das Schlüsselwort `require`, welches zu definierende Bedingungen überprüft und die Ausführung eines Programms abbricht, falls die Bedingungen verletzt sind. Das entwickelte Artefakt nutzt diesen Mechanismus und überprüft mittels `require`, ob alle Bedingungen für eine gegebene Prozessschrittausführung erfüllt sind. Dafür speichern die aus dem Modell extrahierten Requirements für jede Aktivität welche anderen Aktivitäten bereits abgeschlossen sein müssen, um die entsprechende Aktivität ausführen zu können. Die Überprüfung erfolgt dann mittels `require`.

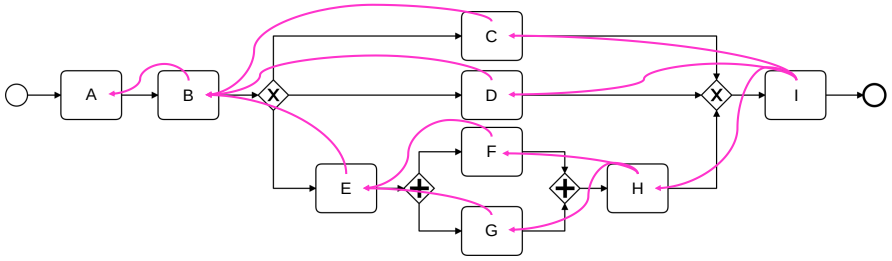

Abbildung 31: Beispielprozess in BPMN mit Requirements (in pink)

Requirements und BPMN Aktivitäten

Tabelle 4 zeigt die Requirements des in Abbildung 31 dargestellten Prozessmodells. Für A sind keine Requirements nötig, da es die erste auszuführende Aktivität ist. Für die Aktivität B wird die Aktivität A als Requirement angegeben, da die beiden Aktivitäten durch einen Sequenzflusspfeil verbunden sind und somit A vor B ausgeführt werden muss. Die Aktivitäten C, D und E folgen dem XOR-Gatter, welches wiederum der Aktivität B nachfolgt. Dementsprechend verlangt eine Ausführung von C, D und E, dass B bereits abgeschlossen ist. Implizit wird somit durch die Transitivität auch gefordert, dass A bereits erledigt ist. Analog definieren die beiden Aktivitäten F und G die Aktivität E als Voraussetzung und die Aktivität H spezifiziert entsprechend F und G als Requirement. Zuletzt können sowohl die Aktivitäten C und D als auch die Aktivität H die Ausführung von I beeinflussen, was sich in den Requirements von I widerspiegelt.

Aktivität	A	B	C	D	E	F	G	H	I
Req.	[]	[A]	[B]	[B]	[B]	[E]	[E]	[F, G]	[C, D, H]
Tasktype	TASK	TASK	TASK	TASK	TASK	TASK	TASK	AND	XOR

Tabelle 4: Extrahierte Requirements des Prozessmodells in Abbildung 31

Requirements und Gatter

Neben den Voraussetzungen in Bezug auf den Lebenszyklusstatus von Aktivitäten sind für die Implementierung der Prozesssemantik

auch die Gatter im BPMN-Modell entscheidend, was über die sogenannten *Tasktypes* abgewickelt wird. Jeder Aktivität ist dabei einer der zur Verfügung stehenden Tasktypes `TASK`, `AND`, `OR` oder `XOR` zugeordnet (Tabelle 4). Die Tasktypes indizieren, ob das vorangegangene Element einer Aktivität eine andere Aktivität oder ein bestimmtes *zusammenführendes* Gatter ist. Die Tasktypes wirken sich wie folgt auf die Überprüfungsroutine aus. Bei einem vorangegangenen parallelen Gatter müssen alle Requirements erfüllt sein, das bedeutet, dass alle Aktivitäten, welche in der Liste an Requirements der auszuführenden Aktivität gelistet sind, abgeschlossen sein müssen. Im Beispiel kann die Aktivität H nur dann ausgeführt werden, wenn sowohl die Aktivität F als auch die Aktivität G bereits ausgeführt sind. Andererseits muss bei einem vorangegangenen exklusiven Gatter nur eine Aktivität aus den Requirements bereits ausgeführt sein. Im Beispiel kann die Aktivität I ausgeführt werden, wenn eine der Aktivitäten C, D oder H bereits ausgeführt ist, was der in BPMN beschriebenen Semantik entspricht. Die genaue Implementierung von diesem Mechanismus wird bei der Vorstellung des Smart Contracts in Kapitel 5.4.2.3 gezeigt.

Tasktypes

5.4.2.2 *Bereitstellung des Modells im Smart Contract*

Der vorherige Abschnitt zeigt, wie die BPMN-Semantik über die Requirements abgebildet werden kann. Im nächsten Schritt ist es notwendig, die Struktur des BPMN-Prozessmodells auf eine Datenstruktur im Smart Contract abzubilden. Dafür muss einerseits eine Transformation des Prozessmodells in diese Datenstruktur erfolgen und andererseits müssen die Informationen in den Smart Contract auf der Blockchain geladen werden. Ein Parser analysiert dabei das in XML serialisierte BPMN-Modell und bestimmt dabei entsprechend der BPMN-Semantik die Requirements und Tasktypes für jede Aktivität. Diese Informationen werden dann mithilfe von Transaktionen an den Smart Contract geschickt und dort gespeichert.

Abbildung der Requirements im Smart Contract

Identifikationsnummer	`uint id;`	8
Aktivitätsname	`string activity;`	H
Ausgeführt-Indikator	`bool completed;`	false
BPMN Element	`Tasktype tasktype;`	AND
Requirements	`uint[] requirements;`	[F, G]

Tabelle 5: Datenstruktur für Aktivitäten am Beispiel der Aktivität H

Tabelle 5 zeigt anhand der Aktivität H, welche Struktur für die Aktivitäten im Smart Contract auf der Blockchain erwartet wird. Neben einer eindeutigen Identifikationsnummer (`id`) wird der Name der Aktivität gespeichert (`activity`). Die Variable `completed` zeigt an, ob die Aktivität bereits ausgeführt ist. Zuletzt werden noch der Tasktype so-

Datenstruktur im Smart Contract

wie die Requirements gespeichert. Für jede einzelne Aktivität wird über jeweils eine Transaktion ein Eintrag in einer Liste im Smart Contract gespeichert. Das folgende Kapitel zeigt, wie diese Datenstruktur für die Prozessausführung genutzt wird.

5.4.2.3 *Struktur und Funktionsweise der Smart Contracts*

Zustandsvariablen im Smart Contract

STRUKTUR DES SMART CONTRACTS Quelltext 4 zeigt die Felder, welche im entwickelten Smart Contract gespeichert werden. Das `struct Task` wird für die Definition der Prozessschritte beziehungsweise der Aktivitäten im BPMN-Modell verwendet. Die einzelnen Felder werden anhand von Tabelle 5 im vorherigen Abschnitt erläutert. Im Smart Contract werden die Requirements als `uint` über die `id` der *Task*s referenziert. Weiterhin wird durch die Verwendung einer booleschen Variable in der aktuellen Implementierung ein einfaches, zweistufiges Lebenszyklusmodell unterstützt (*nicht completed* oder *completed*). Der Datentyp `mapping` wird in Solidity ähnlich einer Hash-Tabelle verwendet. Durch das `mapping(uint => Task)` kann über die `id` der zugehörige Task ermittelt werden. Das Array `uint[] public tasksArray` speichert alle Tasks in einer Liste und der Zahlenwert `uint public taskcount` wird verwendet, um die aktuelle Anzahl an gespeicherten Tasks abzufragen.

```
1  contract ContractCollaborationManager {
2
3      enum Tasktype {TASK, AND, OR, XOR}
4
5      struct Task {
6          uint id;
7          string activity;
8          bool completed;
9          Tasktype tasktype;
10         uint[] requirements;
11     }
12     mapping(uint=>Task) tasks;
13     uint[] public tasksArray;
14     uint public taskcount;
15      ...
16 }
```

Quelltext 4: Strukturelle Elemente des Smart Contracts

addReq zur Bereitstellung (Phase 2)

FUNKTIONEN DES SMART CONTRACTS Für die Bereitstellung des Prozessmodells in den Smart Contract steht die Funktion `addReq` zur Verfügung, welche in Quelltext 5 abgebildet ist. Diese wird für jede Aktivität im ursprünglichen BPMN-Prozessmodell jeweils einmal mit den entsprechenden Parametern aufgerufen. Im Methodenrumpf von `addReq` wird zuerst ein Objekt `struct Task` erzeugt, welches anschließend in der Variable `tasksArray` abgespeichert wird. Die Hilfsvariable `taskcount` unterstützt dabei, die richtige Position im Array zu finden. Diese Aufrufe dienen der Initialisierung des Smart Contracts,

um ihn auf die Verwendung für die Prozessausführung vorzubereiten.

```
1  contract ContractCollaborationManager {
2      ...
3      function addReq(string memory _activity, Tasktype _tasktype, uint[] memory
           _requirements) public {
4          Task storage task = tasks[taskcount];
5          task.id = taskcount;
6          task.activity = _activity;
7          task.completed = false;
8          task.tasktype = _tasktype;
9          task.requirements = _requirements;
10
11         tasksArray.push(taskcount);
12         taskcount = taskcount + 1;
13     }
14
15 }
```

Quelltext 5: Initialisierung des Smart Contracts bezüglich des zu interpretierenden Prozessmodells

setTaskOnCompleted zur Ausführung (Phase 3)

Die anschließende Prozessausführung wird über die Funktion `setTaskOnCompleted` abgewickelt, welche in Quelltext 6 abgebildet ist. Diese Funktion wird von einem Akteur zur Beendigung der Ausführung einer Aktivität aufgerufen. Dabei prüft die Funktion, ob die Beendigung der spezifizierten Aktivität prozesskonform ist, das heißt, ob die vorangegangenen Aktivitäten entsprechend der Requirements unter Berücksichtigung des Tasktypes bereits ausgeführt sind. Dieser Vorgang wird im folgenden Abschnitt erläutert.

Routine zum Ausführen einer Aktivität

In Zeile 5 werden vom auszuführenden Task die als Requirements definierten Tasks für die weitere Verwendung in eine Variable gespeichert. Falls keine Requirements angegeben sind, handelt es sich um den ersten auszuführenden Task, dessen `completed`-Variable somit umgehend auf `true` gesetzt werden kann (Zeilen 7 bis 9). Der darauffolgende Quelltext gliedert sich nach den verschiedenen Tasktypes. Falls der auszuführende Task den Tasktype `TASK` aufweist (Zeile 12), ist das vorherige BPMN-Element ebenfalls eine Aktivität, welche als einzige in den Requirements referenziert ist und auf die Beendigung geprüft werden muss (Zeile 13). Bei einer erfolgreichen Überprüfung wird der auszuführende Task ebenfalls als beendet markiert (Zeile 14), andernfalls wird der Prozessstatus nicht aktualisiert und `false` zurückgegeben (Zeile 16). Anschließend folgt die Logik für die Requirementsüberprüfung von Aktivitäten, welchen ein BPMN-Gatter voransteht. Dies ist am Beispiel des AND-Gatters im Quelltext 6 dargestellt (Zeilen 19 bis 28). Nachdem der Tasktype überprüft ist (Zeile 19), wird in der `for`-Schleife (Zeilen 20 bis 23) in `tempcount` die Anzahl der in den Requirements bereits als abgeschlossen markierten Tasks bestimmt. Basierend auf dieser Anzahl wird nun für die verschiedenen Tasktypes geprüft, ob alle Requirements abgeschlossen

```
1  contract ContractCollaborationManager {
2    ...
3    function setTaskOnCompleted(uint _id) public returns(bool success) {
4      uint tempcount;
5      uint[] memory temprequire = tasks[_id].requirements;
6      //START
7      if(temprequire.length == 0) {
8        tasks[_id].completed = true;
9        return true;
10     }
11     //TASK
12     if(tasks[_id].tasktype == Tasktype.TASK) {
13       if(isTaskCompletedById(temprequire[0]) == true) {
14         tasks[_id].completed = true;
15         return true;
16       } else { return false; }
17     }
18     //GATEWAYS
19     if(tasks[_id].tasktype == Tasktype.AND) {
20       for(uint i = 0; i < temprequire.length ; i++) {
21         if(isTaskCompletedById(temprequire[i])==true) {
22           tempcount++; }
23       }
24       if(tempcount == temprequire.length) {
25         tasks[_id].completed = true;
26         return true;
27       } else { return false; }
28     }
29     ...
30 }
```

Quelltext 6: Routine zur Prüfung der Requirements

sind (für Tasktype `AND` (Zeile 24)) oder ein einziges Requirement erfüllt ist (für Tasktype `XOR`).

In Kapitel 5.6 wird unter anderem für die `setTaskOnCompleted`-Methode die Integration der organisationalen Perspektive anhand der Prüfung von Ausführungsrechten gezeigt.

5.5 DEMONSTRATION

Dieses Kapitel stellt die Verwendung des entwickelten Artefakts vor. Dabei sollen die wichtigsten Fragen und Herausforderungen diskutiert werden, welche beim Einsatz zu beantworten und zu lösen sind. Die Grundstruktur dieses Kapitels geben die in Abbildung 30 dargestellten Phasen der Vorbereitung und Modellierung, Überführung in die Blockchain-Domäne, Bereitstellung sowie der Ausführung von Prozessmodellen vor.

Zum Zeitpunkt des Schreibens dieser Dissertation existiert keine Anwendung mit einer grafischen Benutzerschnittstelle für das vorgestellte Artefakt. Stattdessen kann der Smart Contract zum Beispiel über die browserbasierte Entwicklungsumgebung *Remix*[3] bereitgestellt und über den Aufruf der Funktionen getestet werden. Am Ende des

3 https://remix.ethereum.org/, besucht am 15.02.2024

Kapitels wird die ordnungsgemäße Funktionsweise des Artefakts zusätzlich anhand eines Softwaretests beschrieben.

Phase 0: Blockchain-Infrastruktur bereitstellen und Prozessmodellierung

Bereitstellung der Infrastruktur

In einem ersten Schritt für den praktischen Einsatz muss eine Blockchain-Infrastruktur gewählt, konfiguriert und aufgesetzt werden, auf der die Smart Contracts bereitgestellt werden können. Bereits hier sind unter anderem die folgenden Entscheidungen zu treffen, welche vom Anwendungsfall oder weiteren Faktoren wie den Kosten oder Sicherheitsanforderungen abhängig sein können.

verschiedene EVM-Blockchains

SELEKTION DES BLOCKCHAIN-PROTOKOLLS Obwohl das Ethereum-Protokoll bei diesem DSR-Projekt im Fokus steht, existieren weitere Protokolle, welche die EVM integrieren und in Solidity programmierte Smart Contracts ausführen können.[4,5] Zur Umsetzung einer Blockchain-basierten Prozessausführung stehen somit verschiedene Blockchain-Protokolle zur Verfügung. Die Protokolle unterscheiden sich unter anderem hinsichtlich Durchsatz, Kosten oder Offenlegung der Daten [174] und können sich damit je nach den Anforderungen des speziellen Anwendungsfalls mehr oder weniger gut eignen. Die Frage nach der Eignung eines Protokolls für die Prozessausführung, oder ob sich die jeweilige Domäne des Prozesses auf die Eignung auswirkt, soll hier nicht weiter diskutiert werden. Stattdessen wird im Einklang mit den verwandten Arbeiten, zum Beispiel [53, 110, 185], in dieser Arbeit ebenfalls die Verwendung des Ethereum-Protokolls angenommen.

Wahl der Sichtbarkeit und Konsensmechanismus

KONFIGURATION DER BLOCKCHAIN In Kapitel 4 sind verschiedene Konfigurationsmöglichkeiten der Ethereum-Blockchain beschrieben, welche sich unter anderem in der Sichtbarkeit des Netzwerks beziehungsweise dem Konsensverfahren unterscheiden, also der Wahl der Algorithmen für die Validierung und Synchronisierung von Transaktionen. Die Konfiguration einer Blockchain erfordert dabei stets ein anwendungsspezifisches Ausbalancieren des Trilemmas bezüglich Skalierbarkeit, Dezentralisierung und Sicherheit, weswegen auch hier keine allgemeingültige Lösung für alle Kollaborationen vorgeschlagen werden soll.

unbegrenzte, pseudonymisierte Teilnahme notwendig?

Für die Entscheidungsfindung ist zu prüfen, ob ein Anwendungsfall den Einsatz einer Blockchain erfordert, welche mit einem hochskalierbaren aber kostenintensiven Konsensverfahren wie Proof of Work oder Proof of Stake gesichert ist. Die Vorteile speziell von unbegrenz-

4 https://wiki.polygon.technology/docs/pos/design/bor/core_concepts/#evmsolidity-as-vm, besucht am 03.12.2023

5 https://support.avax.network/en/articles/5417030-what-is-the-ethereum-virtual-machine-evm, besucht am 03.12.2023

ten, pseudonymisierten Benutzerkonten zusammen mit einer starken Dezentralisierung müssen die Nachteile der hohen Transaktionskosten rechtfertigen. Alternative Netzwerke mit weniger skalierbaren Synchronisierungsalgorithmen aber einer erforderlichen Offenlegung der Identitäten der Teilnehmer wie bei Proof or Authority-Verfahren können hingegen ohne den Einsatz von Kryptowährungen, allerdings auf Kosten der Dezentralisierung, überzeugen.

Es sei nochmals angemerkt, dass die Wahl des Protokolls und der Konfiguration unabhängig vom entwickelten Artefakt ist.

Starten des Netzwerks

INITIALISIERUNG UND BETRIEB DES NETZWERKS Sobald sich die Kollaborationsteilnehmer auf ein Protokoll und dessen Konfiguration geeinigt haben, muss das Netzwerk für die Verwendung initialisiert werden. Dies erfordert unter anderem die Erstellung einer Konfigurationsdatei für private Netzwerke, in der zum Beispiel die Adressen der Validierungskonten spezifiziert werden, welche am Mining-Prozess teilnehmen. Weiterhin muss dafür gesorgt werden, dass alle möglichen Prozessakteure ein Benutzerkonto auf der Blockchain besitzen, um Transaktionen signieren und versenden zu können.

Erstellen der Benutzeraccounts

Prozess modellieren

PROZESS MODELLIEREN Während bei der nachrichtenbasierten zwischenbetrieblichen Prozessausführung zum Beispiel BPMN-Choreografien und BPMN-Kollaborationsdiagramme zum Einsatz kommen, wird bei dem vorgestellten Artefakt zur Blockchain-basierten Prozessausführung ein BPMN-Prozessdiagramm verwendet. Die Kollaborationsteilnehmer müssen sich demnach auf ein gemeinsames globales Prozessmodell und gegebenenfalls ein Organisationsmodell einigen, welches den zwischenbetrieblichen Prozess und die Struktur der möglichen Prozessakteure beschreibt. Dies erfolgt ebenfalls unabhängig vom vorgestellten Artefakt und wird hier nicht weiter diskutiert.

Phase 1: Prozess in Blockchain-Domäne überführen

Extraktion der Requirements

In Phase 1 erfolgt die Transformation von BPMN in eine EVM-verständliche Form des Prozesses. In Hinblick auf das in Phase 2 bereitgestellte Artefakt, welches ein BPMN-Prozessmodell nicht direkt in der serialisierten XML-Form interpretieren kann, werden daher zuvor benötigte Requirements extrahiert. Dies kann entweder manuell erfolgen oder über ein Skript, welches ein serialisiertes Prozessmodell einliest und bei der Traversierung der XML-Baumstruktur die entsprechenden Requirements identifiziert.

Phase 2: Bereitstellung

Im nächsten Schritt erfolgt die Bereitstellung des transformierten Prozessmodells. Dabei wird das Blockchain-agnostische BPMN-Prozess-

modell in Form der extrahierten Requirements in der Turing-vollständigen EVM-Umgebung implementiert. Dazu muss der generische Interpreter-Smart Contract über eine Transaktion, deren Sender von den Kollaborationsteilnehmern zu bestimmen ist, auf der Blockchain bereitgestellt werden. Mit der nun bekannten Adresse des bereitgestellten Smart Contracts können nachfolgend die Tasks inklusive den Requirements auf den Smart Contract in die Datenstrukturen eingepflegt werden (Kapitel 5.4). Auch diese Transaktionen müssen von einem zu bestimmenden Verantwortlichen gesendet werden, was auch durch abstimmungsbasierte Mechanismen ersetzt werden kann [84].

Bereitstellen des Smart Contracts

Aufruf der addReq-Funktion

Phase 3: Prozess ausführen

Sobald der Smart Contract und das Prozessmodell darauf bereitgestellt sind, kann die Ausführung des Prozesses starten. Dafür ruft ein berechtigter Prozessteilnehmer die entsprechende Funktion des Smart Contracts inklusive der passenden Parameter auf. Zum Beispiel würde der Funktionsaufruf `setTaskOnCompleted(1)`, welcher das `completed`-Attribut des Tasks mit der `id` `1` auf `true` setzt, erfolgreich durchlaufen und somit den Prozessstatus aktualisieren. Ein erneuter Aufruf der Funktion mit dem Parameter 1 ist nicht prozesskonform und hätte damit keine Auswirkung auf den Prozessstatus. Ähnlich hätte ein Funktionsaufruf, bei dem die Requirementsüberprüfung fehlschlägt, keine Auswirkungen auf den Prozessstatus.

Aufruf der setTaskOnCompleted-Funktion

Beschreibung der Funktionsweise mit einem Softwaretest

In Quelltext 7 ist die Verwendung des Artefakts mit Hilfe eines Software-Tests beschrieben. Zu Beginn wird sowohl der Smart Contract (Zeile 4) sowie die Tasks mit dem Aktivitätsnamen, den Tasktypen und den Requirements darauf bereitgestellt (Zeilen 5 bis 7). Dabei hat B den Task mit der `id` 0 (A) als Requirement und C hat B als Requirement. Die entsprechenden Transaktionen werden vom Benutzerkonto `accounts[0]` gesendet. Im ersten Test (Zeilen 9 bis 19) wird der ordnungsgemäße Ablauf geprüft, indem die `setTaskOnCompleted`-Methode der Reihe nach für die Aktivitäten A, B und C aufgerufen wird. Anschließend wird überprüft, ob die `completed`-Attribute aller Tasks auf `true` gesetzt sind. Im zweiten Test wird ein fehlerhaftes Verhalten simuliert, indem bevor Task A ausgeführt wurde (Überprüfung in den Zeilen 21 und 22), die Ausführung des Tasks B simuliert wird (Zeile 23). Die Zeilen 24 und 25 zeigen anschließend, dass die Prozesskonformität eingehalten wurde und der Aufruf der `setTaskOnCompleted`-Funktion die `completed`-Variable von Task B nicht auf `true` setzte.

Unit-Test für den Smart Contract

```
contract("ContractCollaborationManager", accounts => {
  let instance;
  beforeEach(async () => {
    instance = await ContractCollaborationManager.new();
    await instance.addTask("A", 0, [],  { from: accounts[0] });
    await instance.addTask("B", 0, [0], { from: accounts[0] });
    await instance.addTask("C", 0, [1], { from: accounts[0] });
  });
  it("should run without error when being executed process conform", async () => {
    await instance.setTaskOnCompleted(0, { from: accounts[1] });
    await instance.setTaskOnCompleted(1, { from: accounts[2] });
    await instance.setTaskOnCompleted(2, { from: accounts[3] });
    let tasko = await instance.getTaskById.call(0);
    assert.equal(tasko.completed, true);
    let task1 = await instance.getTaskById.call(1);
    assert.equal(task1.completed, true);
    let task2 = await instance.getTaskById.call(2);
    assert.equal(task2.completed, true);
  });
  it("should not execute B, when A was not executed before", async() => {
    let tasko = await instance.getTaskById.call(0);
    assert.equal(tasko.completed, false);
    await instance.setTaskOnCompleted(1, { from: accounts[2] });
    let task1 = await instance.getTaskById.call(1);
    assert.equal(task1.completed, false);
  }); ...
```

Quelltext 7: Demonstration des Artefakts in einem Softwaretest

5.6 EVALUATION

Dieses Kapitel setzt sich mit der kritischen Evaluation des entwickelten Artefakts gegen die definierten Anforderungen auseinander (Kapitel 5.6.1). Insbesondere wird die Erweiterbarkeit hinsichtlich der organisationalen Perspektive demonstriert, da diese obligatorisch für die Erstellung von Arbeitslisten zur Integration menschlicher Prozessteilnehmer ist. Abschließend fasst Kapitel 5.6.2 diesen DSR-Zyklus zusammen.

5.6.1 *Evaluation der Anforderungen*

A1: VERWENDUNG VON BPMN PROZESSDIAGRAMMEN Basierend auf der Semantik von BPMN-Prozessmodellen werden beim vorgestellten Ansatz sogenannte Requirements bestimmt. Die Requirements bezüglich einer auszuführenden Aktivität spezifizieren, welche anderen Aktivitäten bereits ausgeführt sein müssen und dienen somit zur Überprüfung der Prozesskonformität im Smart Contract.

Semantik über Requirements

Limitationen der Requirements

Obwohl grundlegende Strukturen von BPMN-Prozessmodellen, nämlich Aktivitäten sowie XOR- und AND-Gatter dadurch interpretiert werden können, ist die Implementierung von weiterführenden Konstrukten in diese Architektur nicht trivial. So ist beispielsweise festzustellen, dass die Umsetzung der inklusiven Gatter in der aktuellen Version nicht exakt der vom BPMN-Standard spezifizieren Semantik

entspricht, da nicht konforme Pfade nicht deaktiviert werden [113]. Auch werden BPMN-Events oder nicht block-strukturierte Prozesse aktuell nicht unterstützt [168].

Zusammenfassend können die zwischenbetrieblichen Prozesse als BPMN-Prozessdiagramme ausgeführt werden, sofern bei der Modellierung ausschließlich die unterstützten Elemente verwendet werden. Da innerhalb des interpretierenden Ansatzes die Implementierung der Semantik flexibel ist, können statt der Extraktion von Requirements in künftigen Arbeiten andere Ansätze für die Interpretation von BPMN-Prozessmodellen auf der Blockchain verwendet werden, wie die Integration des Token-Spiels oder Petri-Netze. Das kann die Unterstützung weiterer BPMN-Konstrukte ermöglichen.

Token-Spiel für künftige Arbeiten

A2: VERWENDUNG DER ETHEREUM-BLOCKCHAIN Der Ansatz ist auf Basis eines Smart Contracts umgesetzt, welcher in der Sprache Solidity programmiert ist. Solidity kann für die Verwendung in der EVM kompiliert werden[6], sodass die Verwendung der Ethereum-Blockchain möglich ist. Wie in Kapitel 5.5 bereits diskutiert, ist die Entscheidungsfindung bezüglich der Konfiguration des Blockchain-Netzwerkes allerdings nicht trivial und wird von Faktoren, darunter Performance, Kosten, Skalierbarkeit oder Sicherheit beeinflusst [174].

Solidity-Smart Contract

Konfiguration des Netzwerks

A3: INTERPRETATION NACH WFMS-VORBILD Existierende Ansätze verfolgen einen kompilierenden Ansatz zur automatischen Generierung von spezifischem Smart Contract-Quelltext für Prozessmodelle. Im Gegensatz dazu verwendet der vorgestellte Ansatz einen generischen Smart Contract, welcher zur Ausführung verschiedener Prozessmodelle verwendet werden kann. Die Verwendung erfordert das Bereitstellen des Smart Contracts auf der Blockchain als Analogie zur Installation eines WFMS. Anschließend muss das auszuführende Prozessmodell im WFMS verfügbar gemacht werden, was im vorgestellten Ansatz über die Requirements-Transaktionen umgesetzt ist. Zuletzt kann über eine Methode, welche mit der auszuführenden Aktivität parametrisiert ist, der Prozessstatus aktualisiert werden. In dieser Hinsicht ist das Paradigma der interpretierenden Ausführung nach WFMS-Vorbild umgesetzt.

generischer Smart Contract als WFMS

Allerdings lässt sich der vorgestellte Ansatz nicht als vollwertiges WFMS betiteln. Der Hauptgrund dafür ist die rudimentäre Umsetzung des Aktivitätslebenszyklus. Während bei einer fortgeschrittenen Konzeption des Aktivitätslebenszyklus Phasen wie *bereit*, *zugewiesen*, *gestartet* und *beendet* durchlaufen werden, sind im vorgestellten Ansatz nur die Phasen *bereit* und *beendet* von Aktivitäten umgesetzt. Dies verhindert eine Koordination der Arbeitsschritte, indem das *Starten* der Abarbeitung einer Aktivität nicht koordiniert ist und theoretisch alle

limitierter Aktivitätslebenszyklus

keine fortgeschrittene Koordination der Arbeitsschritte

6 https://docs.soliditylang.org/en/latest/using-the-compiler.html#commandline-compiler, besucht am 04.12.2023

Prozessbeteiligten eine Aktivität starten können. Lediglich die Kommunikation des Abschließens einer Aufgabe wird kommuniziert und auf Prozesskonformität geprüft. In künftigen Arbeiten sind also Funktionalitäten, wie etwa `claim(Task t)` zu implementieren, wofür ein mehrwertiger Datentyp für die Verwaltung des Lebenszyklus der Aktivitäten eingesetzt werden muss.

A4: ERWEITERBARKEIT FÜR WEITERE PROZESSPERSPEKTIVEN Die Erweiterbarkeit des vorgestellten Artefakts wird in [168] hinsichtlich der organisationalen Perspektive und in [169] bezüglich der informationsorientierten Perspektive gezeigt. Quelltext 8 demonstriert eine Erweiterung des Smart Contracts für eine rudimentäre Umsetzung der organisationalen Perspektive.

Erweiterung hinsichtlich der organisationalen Perspektive

```
1  contract ContractCollaborationManager {
2
3      struct Task {
4        ...
5        address resource;
6        WRP wrp;
7      }
8
9      enum WRPType { RDA, RRBA, RSOD }
9
11     struct WRP {
12       WRPType wrptype;
13       address _address;
14       string _role;
15       uint _taskid;
16     }
17
18     struct Collaborator {
19       address resource;
20       string role;
21     }
22     mapping(address => Collaborator) collaborators;
23     address[] public collaboratorArray;
24
25     function addCollaborator(address _collaborator, string memory _role) public {
26       Collaborator storage collaborator = collaborators[_collaborator];
27       collaborator.role = _role;
28       collaboratorArray.push(_collaborator);
29     }
30     ...
31 }
```

Quelltext 8: Erweiterung des Interpreter-Smart Contracts um die organisationale Perspektive

Integration von WRPs

Das `struct Task` wird um eine `address`- und eine `WRP`-Variable ergänzt (Zeilen 5 und 6). Die `address`-Variable speichert den korrespondierenden öffentlichen Schlüssel der Person, welche die Aktivität ausgeführt hat beziehungsweise welche die entsprechende Transaktion signiert hat. Das `struct WRP` referenziert ein sogenanntes *Workflow Resource Pattern (WRP)* [148], um Bedingungen in der organisationalen Perspektive zu spezifizieren. Das WRP ist in der dargestellten Imple-

mentierung entweder vom Typ `RDA`, `RRBA` oder `RSOD`, welche die Muster *direct allocation* (deutsch: direkte Zuweisung), *role-based allocation* (deutsch: rollenbasierte Zuweisung) oder *separation of duties* (deutsch: Vier-Augen-Prinzip) referenzieren (Zeile 9). Zur weiteren Beschreibung des WRP werden die Variablen `_address` (für `RDA`), `_role` (für `RRBA`) oder `_taskid` (für `RSOD`) entsprechend gesetzt (Zeilen 11–16). Die `_taskid` referenziert diejenige Aktivität, deren ausführender Prozessteilnehmer, welcher im `resource`-Attribut des Tasks gespeichert ist, sich vom Sender der Transaktion für die Ausführung der aktuellen Aktivität unterscheiden muss.

Integration eines rudimentären Organigramms

Zur Speicherung eines rudimentären organisationalen Modells werden im `struct Collaborator` den potenziellen Prozessakteuren genauer gesagt deren öffentlichen Schlüsseln eine Rolle zugewiesen (Zeilen 18–21). Ähnlich den Requirements werden die potenziellen Prozessteilnehmer dem Smart Contract als Collaborator über eine `addCollaborator`-Funktion hinzugefügt (Zeilen 25–29).

```
1  contract ContractCollaborationManager {
2    ...
3   function setTaskOnCompleted(uint _id) public returns(bool success) {
4    uint tempcount;
5    uint[] memory temprequire = tasks[_id].requirements;
6    WRPType tempwrp = tasks[_id].wrp.wrptype;
7    // direct allocation
8    if (tempwrp == WRPType.RDA) {
9     require(tasks[_id ]. wrp._address == msg.sender, "RDA violated."); }
10   // role-based allocation
11   else if (tempwrp == WRPType.RRBA) {
12    require(keccak256(abi.encodePacked(tasks[_id].wrp._role)) == keccak256(abi.
          encodePacked(collaborators[msg.sender].role)), "RRBA violated."); }
13   // separation of duties
14   else if (tempwrp == WRPType.RSOD) {
15    require(keccak256(abi.encodePacked(tasks[tasks[_id].wrp._taskid].resource)) != keccak256
          (abi.encodePacked(msg.sender)), "RSOD violated."); }
16  ...
```

Quelltext 9: Implementierung der organisationalen Perspektive in der `setTaskOnCompleted`-Funktion

organisationale Perspektive in der setTaskOn-Completed-Funktion

In der Überprüfungsroutine der Funktion `setTaskOnCompleted` werden entsprechend den WRP-Typen zusätzlich Überprüfungen implementiert (Quelltext 9). Für den Fall, dass der aktuell auszuführenden Aktivität ein WRP vom Typ `RDA` zugeordnet ist, wird in den Zeilen 7 bis 9 überprüft, dass die Senderadresse der aktuellen Transaktion der spezifizierten Adresse in der WRP-Struktur entspricht. Andernfalls wird über das `require`-Schlüsselwort ein Fehler geworfen und die Verarbeitung abgebrochen. Für die Umsetzung des `RRBA`-WRP wird in den Zeilen 10 bis 12 in ähnlicher Weise die im zugewiesenen WRP spezifizierte Rolle mit der Rolle verglichen, welche der aktuellen Senderadresse im Collaborators-Array zugewiesenen ist. Der Vergleich der Zeichenketten erfolgt in Solidity dabei über einen Vergleich der Hashwerte der Zeichenketten. Zuletzt wird in den Zeilen 13 bis 15

Modellierung im BPMN-Modell

das RSOD-Muster implementiert. Hierfür erfolgt ein Vergleich der Senderadresse der aktuellen Transaktion mit der Adresse, die dem im RSOD-WRP spezifiziertem Task in der `resource`-Variable zugewiesenen Adresse. Die Modellierung der direkten oder rollenbasierten Zuweisung sowie der Vier-Augen-Prinzip-Bedingung im BPMN-Modell kann beispielsweise über die RALph-Erweiterung und der Benutzeroberfläche des RALph-Miners erfolgen [20, 21].

Unit-Test für die organisationale Perspektive

Die Funktionsweise beziehungsweise die Überprüfung der korrekten Ausführung ist wiederum anhand eines Beispiels in einem Softwaretest gezeigt (Quelltext 10).

```
1  contract("ContractCollaborationManager", accounts => {
2   let instance;
3   beforeEach(async () => {
4    instance = await ContractCollaborationManager.new();
5    await instance.addCollaborator(accounts[1], "stud", { from: accounts[0] });
6    await instance.addCollaborator(accounts[2], "prof", { from: accounts[0] });
7    await instance.addCollaborator(accounts[3], "stud", { from: accounts[0] });
8    await instance.addTask("A", 0, [],   [0, accounts[1], "", 0], { from: accounts[0] });
9    await instance.addTask("B", 0, [0], [1, accounts[0], "prof", 0], { from: accounts[0] });
10   await instance.addTask("C", 0, [1], [2, accounts[0], "", 0], { from: accounts[0] });
11  });
12  it("should revert, when RSOD constraint is violated", async () => {
13   await instance.setTaskOnCompleted(0, { from: accounts[1] });
14   await instance.setTaskOnCompleted(1, { from: accounts[2] });
15   try {
16    // must ont be executed by accounts[1], who has executed the first task
17    await instance.setTaskOnCompleted(2, { from: accounts[1] });
18    assert.fail('Expected revert not received. Test fails if this line is reached.');
19   } catch (error) {
20    const revertFound = error.message.search('RSOD violated.') >= 0;
21    assert(revertFound, `Expected "RSOD violated.", got ${error} instead`);
22   }
23  }); ...
```

Quelltext 10: Demonstration der Erweiterung bezüglich der organisationalen Perspektive im Softwaretest

Darin wird einem Smart Contract, nach dessen Bereitstellung (Zeile 4), ein Prozessmodell inklusive den Prozessteilnehmern und Rollen zugewiesen (Zeilen 5 bis 11). Der Methode `addTask` wird neben den Parametern für den Aktivitätsnamen, dem Tasktype, der Liste an Requirements auch die Informationen für das WRP, bestehend aus WRP-Typ, Adresse, Rolle und Id eines Tasks übergeben. Im Test (Zeilen 12 bis 22) wird eine fehlerhafte Ausführung in Bezug auf das RSOD-Muster simuliert. Über den Aufruf der `setTaskOnCompleted`-Funktion in Zeile 13 wird dem Task A mit der Id `0` die `resource` des Senderkontos der Transaktion (`accounts[1]`) zugewiesen. Über das WRP des Tasks C wird jedoch spezifiziert, dass C von einer anderen Ressource ausgeführt werden muss. Wenn in diesem Fall eine nicht prozesskonforme Aktion durchgeführt werden soll, zum Beispiel indem für die Ausführung von C derselbe Account verwendet wird, wie für die Ausführung von A, wird ein Fehler erwartet (`"RSOD`

`violated."`), dessen tatsächliches Auftreten in den Zeilen 20 und 21 überprüft wird.

Unabhängigkeit vom Prozessmodell

A5: SKALIERBARKEIT FÜR GROSSE PROZESSMODELLE Die Struktur des Interpreter-Smart Contracts des interpretierenden Ansatzes ist im Gegensatz zum generierten Smart Contract des kompilierenden Ansatzes unabhängig vom jeweiligen Prozessmodell. Damit wächst die Größe des Quelltextes auch nicht mit der Anzahl an Elementen im Prozessmodell. In dieser Hinsicht skaliert der interpretierende Ansatz also besser als der kompilierende Ansatz. Allerdings bleibt zu erwähnen, dass beim vorgestellten Artefakt, die Requirements für die Prozessmodellsemantik über Transaktionen in den Smart Contract geladen werden müssen, wodurch die Skalierbarkeit beeinträchtigt wird. Dies kann allerdings in künftigen Arbeiten verbessert werden, indem die Struktur oder Semantik des Prozessmodells in einer bestimmten Repräsentation innerhalb einer einzigen Transaktion auf den Smart Contract geschrieben wird. Für eine über die deskriptive Analyse hinausgehende detailliertere Evaluation sei auf [111] verwiesen, worin die Skalierbarkeit quantitativ verglichen wird. Die Arbeit wird in Kapitel 8 näher beschrieben.

Anzahl der addReq-Aufrufe

addAllReqs in künftigen Arbeiten

5.6.2 *Zusammenfassung*

Zusammenfassend wird mit dem hier präsentierten Artefakt die interpretierende Prozessausführung auf der Ethereum-Blockchain als Alternative zum bereits vorgestellten kompilierenden Ansatz eingeführt. Die interpretierende Ausführung nach WFMS-Vorbild ermöglicht aus konzeptioneller Sicht flexible Entscheidungen oder dynamische Änderungen am Prozess- und Organisationsmodell. Das vorgestellte Artefakt stellt dabei eine rudimentäre Implementierung dar, die zumindest grundlegende Konstrukte aus der funktionalen, verhaltensorientierten und organisationalen Perspektive unterstützt. Obwohl das Artefakt damit die interpretierende Ausführung demonstriert, werden in der Evaluation weitere Herausforderungen identifiziert: Der Verwaltung des Prozessstatus mit Requirements ist zum Beispiel eine Token-basierte Strategie vorzuziehen, um weitere BPMN-Konstrukte einfacher abbilden zu können. Außerdem ist in den nächsten Schritten der binäre Aktivitätslebenszyklus um feingranularere Stufen zu erweitern, sodass Ausführungsrechte von menschlichen Prozessbeteiligten beansprucht werden können. Da die angemerkten Verbesserungen größtenteils eine weiterführende Neuimplementierung von WFMS-Funktionalitäten erfordern, wird im nächsten DSR-Zyklus stattdessen eine Architektur zur Anbindung von bestehenden WFMSs gefordert (Kapitel 6).

Teil III

EIN RAHMENWERK FÜR DIE FLEXIBLE KONFIGURATION EXTERNER SYSTEME ZUR DEZENTRALEN PROZESSKONTROLLDATENVERWALTUNG

6

DPEX: EIN RAHMENWERK ZUR DEZENTRALEN PROZESSAUSFÜHRUNG

In Kapitel 5 werden Blockchain-basierte Lösungen zur zwischenbetrieblichen Prozessausführung vorgestellt. Die diskutierten Ansätze weisen jedoch eine starke Kopplung zwischen der Prozessausführung und der Blockchain als implementierende Infrastruktur auf, da die Logik direkt über Smart Contracts in der Blockchain-Domäne umgesetzt ist. Dies führt zu einer hohen Kohärenz der Module, welche dadurch schwer austauschbar oder wartbar sind und sich schwer unabhängig voneinander weiterentwickeln lassen. Um dieser Kopplung entgegenzuwirken, wird im folgenden Kapitel ein Middleware-basiertes Schichtenmodell mit klar definierten Schnittstellen und eine darauf aufbauende Referenzarchitektur eingeführt. Die Middleware kann einerseits existierende WFMSs für die Interpretation von Prozessmodellen anbinden und verbindet sich andererseits unabhängig davon zu Kommunikationsinfrastrukturen zur sicheren Nachrichtensynchronisierung. Letztere werden im Folgenden als Secure Communication Infrastuctures (SCIs) (deutsch: sichere Kommunikationsinfrastruktur) eingeführt, wozu auch die in Kapitel 4 vorgestellten Blockchains zählen. Die Middleware stellt dabei die notwendige Kommunikation der angebundenen Komponenten (WFMS und SCI) untereinander sicher.

Die in Kapitel 5 identifizierten Schwachstellen werden im Folgenden in die Problembeschreibung eines zweiten DSR-Zyklus aufgenommen. Dabei gibt Kapitel 6.1 zuerst einen Überblick über das dpex-Framework, bevor Kapitel 6.2 die Problemstellungen der Blockchain-basierten Ansätze beschreibt. Im Anschluss listet Kapitel 6.3 die Anforderungen an ein DSR-Artefakt zur Lösung der Herausforderungen auf. Das Artefakt wird als konzeptionelle Architektur in Kapitel 6.4 vorgestellt. Kapitel 6.5 demonstriert die Verwendung des Artefakts, indem die konzeptionelle Architektur im Zuge einer Machbarkeitsstudie implementiert wird. Insbesondere wird demonstriert, wie sich das WFMS Camunda und die Ethereum-Blockchain in dpex zur dezentralen Prozessausführung integrieren und bezüglich der organisationalen Perspektive und der probabilistischen Finalität erweitern lassen. Zuletzt erfolgt in Kapitel 6.6 eine Evaluation, ob sich die Anforderun-

gen über die konzeptionelle Architektur am Beispiel der gezeigten Implementierung umsetzen lassen.

6.1 EINFÜHRUNG IN DPEX

Das Kapitel stellt den Middleware-basierten Ansatz zur dezentralen Ausführung von Prozessen (englisch: ***d**ecentralized **p**rocess **ex**ecution*), kurz dpex, vor. dpex wird als Artefakt im zweiten Zyklus des DSR-Projekts entwickelt. Um einen Eindruck von der Funktionsweise des Frameworks zu vermitteln, beschreibt dieses Kapitel die verschiedenen Facetten der Implementierung von dpex inklusive deren Verwendung.

Ähnlich wie bei der in Kapitel 5 vorgestellten Blockchain-basierten Prozessausführung liegt der Fokus von dpex auf der dezentralen Ausführung von Prozessen und fällt damit in den Bereich von Decentralized Control [169].

gesamtheitlicher Ansatz

globales Prozessmodell

WFMS-Funktionalitäten

global akzeptierter Datenstand

Unabhängigkeit

Modularität

DPEX ALS FRAMEWORK In erster Linie ist dpex als Framework anzusehen. Als Framework bietet dpex einen gesamtheitlichen Ansatz, wie die zwischenbetriebliche Prozessausführung ohne zentrale Entität zur Koordination umgesetzt werden kann. Auf Ebene der Modellierung wird ein globales Prozessmodell für die Beschreibung der Kollaboration vorausgesetzt statt einer Spezifikation des erwarteten Nachrichtenaustauschs, wie es teilweise bei alternativen Ansätzen verwendet wird. In dieser Hinsicht fokussiert sich dpex ähnlich wie der DSR-Zyklus aus Kapitel 5 auf die Ausführung dieses globalen, zwischenbetrieblichen Prozessmodells nach innerbetrieblichem Vorbild. Das bedeutet, dass alle Vorteile und Funktionen, welche ein gängiges WFMS bei der zentralisierten Prozessausführung bietet, erhalten bleiben sollen. Dazu zählen das Überwachen des Prozessfortschritts, die Sicherstellung der Prozesskonformität während der Ausführung oder das Bereitstellen von Aufgabenlisten für Prozessbeteiligte. Insbesondere sei hier auch auf die Umsetzung der gängigsten Prozessperspektiven verwiesen (Kapitel 2.5).

Die grundsätzliche Herausforderung dabei ist, einen global akzeptierten Datenstand zu schaffen, um einen virtuellen, zentralen Koordinator zu simulieren. Dabei spezifiziert dpex sowohl eine Referenzarchitektur wie solche Systeme entwickelt werden können als auch wie die Infrastruktur und der generelle Ablauf einer Kollaboration umzusetzen sind. Im Gegensatz zu bisherigen Lösungen bleibt die vorgeschlagene abstrakte Referenzarchitektur dabei unabhängig von bestimmten Modellierungssprachen, Ausführungsparadigmen, Synchronisierungsalgorithmen oder Programmiersprachen.

DPEX ALS PLATTFORM dpex bietet eine Plattform, worin verschiedene Module für die Implementierung bestimmter Aufgaben inte-

griert werden können und unabhängig voneinander austauschbar sind. Dabei sind Module einerseits für die BPM-Domäne zur Interpretation von Prozessmodellen bereitzustellen und andererseits für die Kommunikation, Synchronisierung und Sicherheit des Nachrichtenaustausches in der SCI-Domäne. Über einen Plug-In-Mechanismus können die Module in der dpex-Plattform integriert werden, wobei die Plattform sowohl die Schnittstellen dafür bereitstellt als auch die Kommunikation zwischen diesen Modulen während der Laufzeit organisiert. Dadurch können folgende Vorteile erreicht werden:

Plug-in-Mechanismus durch Schnittstellen

- dpex kann für verschiedene Modellierungssprachen und Blockchains beziehungsweise SCIs konfiguriert werden, während im Gegensatz dazu die meisten Blockchain-zentrierten Ansätze lediglich jeweils eine bestimmte Modellierungssprache auf einer bestimmten Blockchain-Plattform unterstützen.
- dpex erlaubt die zeitgleiche Verwaltung mehrerer Kollaborationen innerhalb einer Anwendung.
- Über dpex können die Funktionalitäten existierender Systeme erweitert werden.

Als Beispiel kann einerseits ein Adapter implementiert werden, der ein externes Camunda WFMS zur Interpretation von Prozessmodellen anbindet, und andererseits kann ein Adapter implementiert werden, der die Ethereum-Blockchain zur sicheren Nachrichtensynchronisierung anbindet. Die Adapter werden dann zur Laufzeit der dpex-Anwendung dazu verwendet, Kollaboration zu konfigurieren. In Kapitel 6.5 wird darüber hinaus demonstriert, wie eigene Komponenten für die BPM- oder SCI-Domäne entwickelt werden können und wie bestehende externe Systeme wie Camunda oder Ethereum erweitert werden können.

Erweiterungsmechanismus der Adapter

DPEX ALS LIBRARY Zuletzt existiert dpex in Form einer *Library* (deutsch: Bibliothek). Eine Bibliothek implementiert und kapselt bestimmte Funktionalitäten, welche dann in Softwaresystemen als Black Box eingebunden und verwendet werden können, ohne dass deren genaue Verwendung in einem größeren Kontext vorgegeben ist. In diesem Sinne wird dpex auf Basis von Java als Spring Boot-Bibliothek[1] implementiert, welche dann in Zielprojekten als Abhängigkeit eingebunden werden kann, um ein System nach Vorbild des dpex-Frameworks zu entwickeln.

dpex als Programmbibliothek

WAS IST DPEX NICHT? Wie bereits angedeutet, ist dpex kein Artefakt, keine Software und kein Produkt, womit Prozesse direkt ausgeführt werden können. Das dpex-Konzept stellt lediglich die Middleware bereit, welche für sich weder Prozesse interpretieren noch

Middleware zur Integration externer Komponenten

1 https://spring.io/, besucht am 11.04.2024

Nachrichten austauschen oder synchronisieren kann. Dafür müssen stets Konnektoren zu externen BPM- und SCI-Systemen an die Middleware angebunden werden. Wenn die dpex-Library in einer Spring Boot-Anwendung als Abhängigkeit definiert wird, ermöglicht dies die Implementierung der notwendigen Schnittstellen von dpex in der Anwendung, um externe Systeme anzubinden. Während der Laufzeit verarbeitet dpex dann die Benutzeraktionen und integriert die externen Anwendungen zur dezentralen Prozessausführung. Auf diese Weise können mit dpex Artefakte zur dezentralen Prozessausführung erstellt werden, indem programmatisch die Adapter oder Konnektoren zur Anbindung externer Systeme implementiert und Kollaborationen damit konfiguriert werden.

Vorbereitungen

WIE WIRD DPEX EINGESETZT? Als Voraussetzungen für den Einsatz von dpex müssen sich die Teilnehmer einer zwischenbetrieblichen Kollaboration sowohl auf ein globales Prozessmodell in einer beliebigen Modellierungssprache als auch auf ein bestimmtes Protokoll oder einen Algorithmus hinsichtlich einer SCI einigen. Des Weiteren sind für die Interpretation des Prozessmodells und für die SCI die notwendigen Vorbereitungen zu treffen. Dies beinhaltet die Bereitstellung externer Systeme, zum Beispiel Camunda sowie einen Ethereum-Client, oder eine manuelle Implementierung der entsprechenden Funktionalitäten.

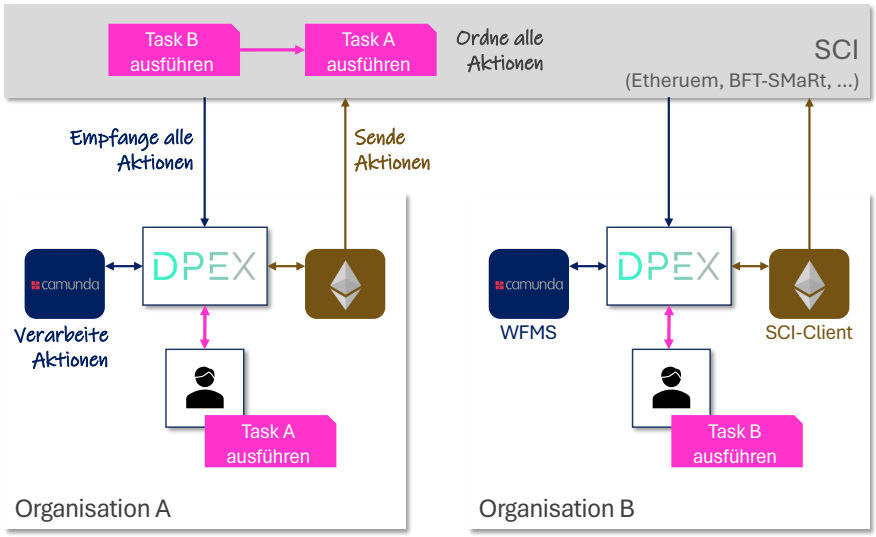

Abbildung 32: Einführung in dpex

Infrastruktur beim Einsatz von dpex

Die Infrastruktur beim Einsatz von dpex ist in Abbildung 32 dargestellt. Jede teilnehmende Organisation verwaltet lokal eine dpex-Instanz, inklusive den lokalen BPM- und SCI-Komponenten. Für dpex wird dann die Kollaboration entsprechend den Abmachungen konfiguriert, das heißt, die Kollaboration wird in der dpex-Anwendung bezüglich der Verwendung der lokalen Komponenten konfiguriert.

Wenn beispielsweise die Kollaboration den zwischenbetrieblichen Prozess als BPMN-Modell definiert, dann muss an die dpex-Middleware ein WFMS angebunden werden, welches BPMN-Prozesse intern interpretieren kann. Wenn Ethereum als SCI verwendet werden soll, wird ein lokal laufender oder extern gehosteter Ethereum-Client angebunden.

Ablauf der Prozessausführung

Während der Prozessausführung sorgt die Middleware nun dafür, dass Benutzeraktionen an das Ethereum-Netzwerk zur Kommunikation mit anderen Teilnehmern geschickt werden und dass möglicherweise konkurrierende Anfragen synchronisiert werden. Alle SCI-Ereignisse, also auch die eigenen Benutzeraktionen, werden anschließend bei einer erfolgreichen Überprüfung von der Middleware an das angebundene WFMS zum Aktualisieren des Prozessstatus weitergereicht. Auf diese Weise verwaltet jeder Teilnehmer lokal den aktuellen Prozessstatus in einem WFMS, während eine SCI dafür sorgt, dass alle Teilnehmer alle Ereignisse in der gleichen Reihenfolge verarbeiten und somit zu jedem Zeitpunkt den gleichen Prozessstatus ableiten können.

6.2 PROBLEMSTELLUNG

Dieses DSR-Projekt bewegt sich im Rahmen der Grundannahmen des Decentralized Control-Prinzips, das heißt, dass verschiedene organisationale Entitäten basierend auf einem globalen, interorganisationalen Prozessmodell zusammenarbeiten. Dabei ist kein zentral verwaltetes WFMS für die Ausführung des Modells verfügbar.

Eine Möglichkeit für die Umsetzung ist in Kapitel 5 mit der Blockchain-basierten Prozessausführung vorgestellt. Bei der Durchführung des DSR-Zyklus sind in Kapitel 5.6 allerdings Probleme identifiziert, welche hier als Motivation für den zweiten DSR-Zyklus dienen. Die folgende Auflistung fasst die Problemstellung und Motivation für diesen DSR-Zyklus zusammen.

existierende Bibliotheken oder Systeme zur Interpretation von Prozessen

RE-IMPLEMENTIERUNG VON WFMSS Bei Blockchain-basierten Lösungen werden die Prozessausführungsfunktionalitäten, welche üblicherweise von WFMSs bereitgestellt werden, von Grund auf in Smart Contracts neu implementiert. In der Softwareentwicklung ist die Duplizierung von Quelltext prinzipiell zu vermeiden. Statt global benutzte Funktionalitäten mehrfach zu implementieren, werden diese üblicherweise über externe Bibliotheken oder Dienste in ein Softwaresystem eingebunden, wodurch sich der Aufwand für die Implementierung und Wartung dieser Funktionalität reduziert und somit Kosten eingespart werden können. Außerdem ist die Wahrscheinlichkeit für Softwarefehler bei ausgereiften, vielfach verwendeten Systemen oder Bibliotheken geringer als bei neu entwickelten Modulen.

Separation von BPM und SCI

KOHÄRENZ DER KOMPONENTEN Ein weiteres Designprinzip aus der Softwareentwicklung ist die Vermeidung einer starken Kopplung von Komponenten oder Modulen. Dies wird dadurch erreicht, dass logisch zusammengehörende Funktionalitäten innerhalb abgeschlossener Module implementiert werden. Wird dieses Designprinzip missachtet, sind bei der Umsetzung von neuen oder geänderten Anforderungen große Teile der Software betroffen, wobei mit einer gut umgesetzten Modularisierung nur die entsprechenden Komponenten geändert und neu getestet werden müssen.

Kohärenz bei Blockchain-basierten Lösungen

Bei der Blockchain-basierten Prozessausführung entsteht durch die Implementierung auf Basis von Smart Contracts eine hohe Kohärenz zwischen der Prozessausführungsfunktionalität und der Kommunikation beziehungsweise Synchronisierung der Nachrichten. Daraus ergeben sich folgende Nachteile. Zum einen steigen bei bestimmten Blockchain-Protokollen und deren Konfigurationen die Kosten, wenn fortgeschrittene Konzepte des Prozessmanagements integriert werden sollen. Denn dadurch steigt die Größe des Quelltextes der implementierenden Smart Contracts und die Ausführungskosten steigen analog zur Größe und Komplexität der Funktionen [189, Formel (114)]. Andererseits kann auch die Integration der Prozessausführung in die IT-Landschaft des Unternehmens aus Sicht der operationalen Perspektive durch die Abgeschlossenheit der Blockchain-Welt eine Hürde darstellen [127]. Dies betrifft beispielsweise Prozessschritte, welche über eine API eine externe Anwendung aufrufen sollen.

Unflexibilität bei Blockchain-basierten Lösungen

PROTOKOLL- UND SPRACHABHÄNGIGKEIT Bei den Artefakten der Blockchain-basierten Prozessausführung (Kapitel 5) besteht eine starke Abhängigkeit sowohl zur verwendeten Modellierungssprache als auch zur ausführenden Infrastruktur. Das vorgestellte Artefakt kann nur BPMN-Prozesse interpretieren und ausschließlich auf EVM-unterstützenden Blockchains eingesetzt werden. Dies hat zur Folge, dass alle Änderungen an externen Protokollen und Sprachen Einfluss auf das Artefakt und damit auf die Prozessausführung haben. Zum Beispiel wird die Programmiersprache Solidity ständig weiterentwickelt, wobei eine Abwärtskompatibilität durch sogenannte *Breaking Changes* nicht gewährleistet ist.[2] Dadurch können Anpassungen an den Smart Contracts der Artefakte erforderlich werden. Außerdem obliegen Blockchain-Protokolle selbst stetigen Änderungen [8, Kap. Appendix B], zum Beispiel die Umstellung von Proof of Work auf Proof of Stake bei Ethereum[3], was einen erheblichen Einfluss auf die Prozessausführung haben kann. Dabei sind dpex-Anwendungen, welche Ethereum-Konnektoren verwenden, genauso betroffen wie die reinen Blockchain-basierten Lösungen. Allerdings sind einerseits die sehr

Einfluss und Abhängigkeit von externen Komponenten

schwächere Abhängigkeit von dpex

2 https://docs.soliditylang.org/en/latest/080-breaking-changes.html, besucht am 07.12.2023

3 https://eips.ethereum.org/EIPS/eip-3675, besucht am 08.12.2023

leichtgewichtigen Smart Contracts (englisch: *Thin Contract*) bei dpex-Anwendungen einfacher zu warten und umzustellen als ein Smart Contract, der die komplette Logik für die Prozessausführung implementiert (englisch: *Fat Contract*). Damit sind die Auswirkungen bei extrinsischen Änderungen beim Einsatz von Thin Contracts nicht so schwerwiegend. Andererseits können durch die klare Schichtentrennung bei dpex die Kollaborationen auch unkompliziert mit anderen kompatiblen Protokollen konfiguriert werden, sodass ein Umstieg bei ungewollten Protokolländerungen einfach umzusetzen ist.

unkomplizierter Austausch von Komponenten

ERWEITERBARKEIT Oftmals kann der Funktionsumfang von gängigen WFMSs den Anforderungen einer fortgeschrittenen Prozessausführung nicht nachkommen. Dies ist am Beispiel der organisationalen Perspektive zu sehen, welche in BPMN-Prozessdiagrammen und damit auch in BPMN-konformen Ausführungssystemen nur rudimentär behandelt wird. Aus diesem Grund ist es wichtig, dass bestehende Systeme leicht erweitert werden können, zum Beispiel um die Funktionalität zur Koordination von Aufgaben zu Mitarbeitern. Während dies bei Blockchain-basierten Ansätzen ebenfalls im Rahmen der Fat Contracts erfolgen muss, erlaubt dpex, dass diese Aspekte in Java und damit außerhalb des geschlossenen Blockchain-Ökosystems unkompliziert implementiert werden können.

unflexible Erweiterbarkeit von Blockchain-basierten Lösungen

VERWALTUNG MEHRERER KOLLABORATIONEN Im allgemeinen Fall kann eine organisationale Entität zu mehreren Kollaborationspartnern zwischenorganisationale Verbindungen unterhalten. Die Kollaborationen können unterschiedliche Anforderungen definieren, sodass verschiedene Prozessmodellierungssprachen zum Einsatz kommen oder andere Implementierungen beziehungsweise Konfigurationen von SCIs bevorzugt werden. Wenn spezifische Artefakte für verschiedene Kollaborationen verwaltet werden müssen, verstärkt das die oben aufgeführten Probleme noch einmal. Das dpex-Framework bietet durch die Möglichkeit zur Verwaltung mehrerer Kollaborationen eine einheitliche Schnittstelle und ermöglicht, Artefakte kollaborationsübergreifend wiederzuverwenden.

Verwaltung mehrerer Kollaborationen

6.3 ANFORDERUNGEN AN DAS DPEX-FRAMEWORK

Das folgende Kapitel definiert die Anforderungen an das dpex-Framework als Artefakt dieses DSR-Zyklus zur dezentralen Ausführung von Prozessen. Die Anforderungen leiten sich aus der Problembeschreibung von Kapitel 6.2 ab und gliedern sich in die Bereiche Prozessmanagement (BPM), sichere Kommunikationsinfrastrukturen (SCI) und Systementwurf.

6.3.1 *BPM-Anforderungen*

Vielfalt von Sprachen

A1: FLEXIBILITÄT HINSICHTLICH PROZESSMODELLIERUNGSSPRACHEN Im BPM-Forschungsbereich werden viele verschiedene Sprachen zur Modellierung von Prozessen vorgestellt, darunter die Diagramme des BPMN-Standards [135], EPKs [118], DECLARE [139], DCR-Graphen [71] oder YAWL [200]. Hinsichtlich der Ausführung fokussieren die Sprachen vorrangig innerbetriebliche Prozesse, welche in einem zentral verwalteten WFMS interpretiert werden. Andererseits werden für zwischenbetriebliche Prozesse unter anderem auch nachrichtenbasierte Konzepte wie Let's Dance [195] oder BPMN-Choreografien vorgeschlagen.

Im Rahmen der Blockchain-basierten Prozessausführung werden verschiedene Konzepte und Sprachen für die Definition zwischenbetrieblicher Prozesse verwendet. Dazu zählen auch YAWL [5], deklarative DCR-Graphen [115] oder die Kollaborationsdiagramme [84], Prozessdiagramme [53, 170] und nachrichtenbasierte Choreografien [27, 96] aus dem BPMN-Standard. Dies zeigt, dass zum aktuellen Zeitpunkt kein allgemein anerkanntes Konzept als Goldstandard zur Umsetzung zwischenbetrieblicher Prozesse existiert. Um die dezentrale Ausführung von verschiedenartig notierten Prozessen zu unterstützen, wird die Unabhängigkeit des Frameworks von konkreten Sprachen oder Konzepten als erste Anforderung definiert. Mit dpex sollen Prozesse interpretiert und ausgeführt werden können, welche in verschiedenen Modellierungssprachen spezifiziert sind. Nachrichtenbasierte Kollaboration sollen von dpex vorerst nicht unterstützt werden. Grund dafür ist die Möglichkeit der Integration unterschiedlicher Informationen bei Prozessmodellen durch die Perspektiven und die dadurch ermöglichte kollaborationsübergreifende Koordination von Arbeitsschritten (Kapitel 3.3).

Unterstützung der Vielfalt in dpex

Limitierte WFMS-Funktionalitäten in Blockchain-basierten Ansätzen

A2: UNTERSTÜTZUNG GÄNGIGER WFMS-FUNKTIONALITÄTEN In den ersten Ansätzen zur Blockchain-basierten Prozessausführung ist der unterstützte Funktionsumfang noch deutlich eingeschränkt im Vergleich zu den weitverbreiteten WFMSs, welche bei der zentralisiert organisierten Prozessausführung im Einsatz sind. Die Überwachung des Prozessstatus ist durch die Verwaltung auf der Blockchain konzeptionell zwar möglich, allerdings ist dies je nach den verwendeten Datenstrukturen im Smart Contract ohne weitere Verarbeitung und Visualisierung nicht transparent und benutzerfreundlich. Über das Requirementsmodell in [170] kann beispielsweise lediglich die Konformität einer neuen Aktion überprüft werden. Die Ableitung des aktuellen Prozessstatus oder der nächsten auszuführenden Aktivitäten ist hingegen nicht direkt möglich. Aus diesem Grund ist zusammen mit der nur rudimentären Umsetzung der organisationalen Perspektive (Anforderung A3) auch eine Implementierung von perso-

nalisierten Arbeitslisten umständlich. Zuletzt ist auch die Bereitstellung eines Ereignisprotokolls zu Analysezwecken oft nicht gegeben und muss aufwendig aus der Blockchain extrahiert werden [86, 128]. Anforderung A2 fasst diese Punkte zusammen und definiert die Unterstützung von gängigen WFMS-Funktionen als Anforderung an das dpex-Framework. Dabei sind vornehmlich die Überwachung des Prozessstatus sowie die Bereitstellung von personalisierten Arbeitslisten und Ereignisprotokollen zu Analysezwecken zu nennen.

Bereitstellung der WFMS-Funktionalitäten im dpex-Framework

A3: UNTERSTÜTZUNG DER GÄNGIGEN PROZESSPERSPEKTIVEN Ansätze zur Blockchain-basierten Prozessausführung sorgen prinzipiell durch die Abbildung der einzuhaltenden Prozesslogik auf Smart Contracts durch eine konforme Abarbeitung. Dabei steht der Kontrollfluss stark im Fokus [110, 170]. Erst in aufbauenden Veröffentlichungen werden weitere Perspektiven prototypisch hinzugefügt, ohne dabei fortgeschrittene Konzepte zu integrieren. Zum Beispiel wird in der organisationalen Perspektive die Definition von Ausführungsrechten über Referenzen zu einem expliziten Organisationsmodell häufig nicht unterstützt [113, 170]. Auch die informationsorientierte Perspektive ist bei manchen Ansätzen nur rudimentär integriert, zum Beispiel durch die manuelle Definition von Prozessvariablen und deren Verwendung bei datenbasierten Entscheidungen [169]. In Kapitel 2.5 werden die wichtigsten Prozessperspektiven vorgestellt. Anforderung A3 postuliert für das dpex-Framework die Möglichkeit der Unterstützung dieser Perspektiven, insbesondere der organisationalen, informationsorientierten und operationalen Perspektive während der Modellierung und Ausführung. Dabei sollen fortschrittliche Konzepte zum Einsatz kommen, zum Beispiel die Verwendung eines expliziten Organisationsmodells.

Ungenügende Unterstützung der Prozessperspektiven bei Blockchain-basierten Ansätzen

Unkomplizierte Integration und Unterstützung der Prozessperspektiven

A4: VERWALTUNG MEHRERER KOLLABORATIONEN Es ist nicht auszuschließen, dass ein Unternehmen an der Prozessausführung mehrerer Kollaborationen beteiligt ist. Während beim Ansatz in Kapitel 5 für jede Prozessinstanz ein neuer, unabhängiger Smart Contract bereitgestellt wird, soll das dpex-Framework eine einheitliche Gesamtverwaltung über alle dezentral ausgeführten Prozesse ermöglichen.

Verwaltung mehrerer Kollaborationen

6.3.2 *SCI-Anforderungen*

Es ist gezeigt, dass die Blockchain-Technologie als Infrastruktur für die dezentrale Prozessausführung eingesetzt werden kann [113, 170]. Die Verwendung der Blockchain ohne genauere Evaluation der Anforderungen aus der BPM-Domäne ist jedoch kritisch zu hinterfragen [166]. Es existieren alternative Protokolle und Systeme, welche sich ebenfalls und gegebenenfalls in bestimmten Anwendungsfällen sogar besser für die dezentrale Prozessausführung eignen, zum Bei-

Blockchain als ideale Lösung?

spiel BFT-Algorithmen [44]. Angesichts dessen werden die Anforderungen aus den wichtigsten Funktionsbausteinen der Blockchain abgeleitet, hier aber unabhängig von einem bestimmten Konzept, System oder Algorithmus als Anforderung an das dpex-Framework definiert.

Kommunikation aller Ereignisse

B1: KOMMUNIKATION Die Kommunikation erfolgt in der Ethereum-Blockchain über ein bestimmtes Gossip-Protokoll, welches für die Verteilung von Transaktionen im gesamten Netzwerk sorgt [83]. Alle Prozessaktionen werden über Transaktionen abgebildet, sodass Ethereum automatisch für die Kommunikation dieser Ereignisse zwischen allen Prozessteilnehmern sorgt. Das dpex-Framework muss ebenfalls eine Verteilung der Information über neue Prozessaktionen ermöglichen.

Globale Ordnung der Ereignisse

B2: SYNCHRONISIERUNG Da eine über verteilte Netzwerkknoten hinweg synchronisierte Reihenfolge aller Nachrichten nicht ohne Weiteres gewährleistet werden kann, kommen dafür bestimmte Algorithmen zum Einsatz. Mit der Bitcoin-Blockchain ist dabei ein innovativer Ansatz vorgestellt, welcher eine prinzipiell unbegrenzte und pseudonymisierte Teilnahme im Netzwerk erlaubt. Allerdings sind dies möglicherweise keine Anforderungen im Bereich der dezentralen Prozessausführung, sodass alternative Algorithmen angewandt werden können. Dadurch können bestimmte Nachteile von Blockchain-Systemen, darunter der Einsatz von Kryptowährungen oder übermäßiger Rechenaufwand, vermieden werden [166]. Als Alternative können sogenannte BFT-Algorithmen integriert werden [44], welche ebenfalls für die Synchronisierung der Netzwerkknoten sorgen, und dabei auch robust gegenüber einem gewissen Maß an Falschinformationen sind [199].

unbegrenzte, pseudonymisierte Benutzer

BFT-Algorithmen

B3: SICHERHEIT Aufgrund der Komplexität dient dieser Punkt nicht einer vollständigen Auflistung aller Sicherheitsanforderungen an das dpex-Framework oder an implementierende Anwendungen. Stattdessen sollen die wichtigsten Anforderungen aufgenommen werden, welche auch im Speziellen bei Blockchain-basierten Lösungen diskutiert sind.

Authentifizierung

Die Authentifizierung von Nutzern ist eine essenzielle Funktionalität bei Anwendungen. Bei Blockchain-basierten Systemen erfolgt die Authentifizierung über asymmetrische Kryptosysteme. Ähnliche Konzepte sollen auch im dpex-Framework integrierbar sein.

Robustheit gegenüber Manipulationsversuchen

Zu den Sicherheitsaspekten zählt auch die Robustheit gegenüber Manipulationsversuchen. Diese werden insbesondere bei verteilten Systemen [99] und Blockchain-Anwendungen oft herausgestellt [115, 185]. Während für Blockchains durch die hochskalierbare und pseudonymisierte Teilnahme innovative Lösungen entwickelt wurden, kön-

nen für Netzwerke mit begrenzter und bekannter Teilnahme auch traditionelle BFT-Algorithmen eingesetzt werden [180]. Ein praktischer Ansatz wurde mit der *State machine replication* (deutsch: replizierte Zustandsautomaten) vorgestellt [98]. Die Idee wurde in [23] weiterentwickelt bis mit BFT-SMaRt eine benutzerfreundliche Implementierung vorgestellt wurde [12], welche beispielsweise für den Einsatz in dpex untersucht werden kann.

BFT-SMaRt

6.3.3 *Anforderungen bezüglich des Systementwurfs*

Die Forderung nach einer flexiblen und leicht erweiterbaren Systemarchitektur wird anhand der identifizierten Probleme von Ansätzen innerhalb der rein Blockchain-basierten Prozessausführung abgeleitet.

Austausch von BPM- und SCI-Komponenten

C1: FLEXIBILITÄT Im Bereich der Blockchain-basierten Prozessausführung existieren für verschiedene Prozessmodellierungssprachen und für verschiedene Blockchain-Protokolle jeweils unterschiedliche Artefakte, welche die Funktionalitäten der Prozessausführung von Grund auf neu implementieren. Um die Notwendigkeit beziehungsweise den Aufwand zur Entwicklung derartiger Artefakte zu reduzieren, soll das in diesem DSR-Zyklus entwickelte Artefakt flexibel bezüglich Prozessmodellierungssprachen (Anforderung A1) aber auch bezüglich Blockchain-Protokollen beziehungsweise SCIs sein. Insbesondere sollen damit sowohl die verwendbaren Prozessmodellierungssprachen vor allem aber die Systeme und Algorithmen der SCI-Domäne leicht austauschbar sein. Zudem soll sichergestellt sein, dass der Austausch von bestimmten Modulen unabhängig von anderen Teilen des Frameworks erfolgen kann.

Unterstützung notwendiger Anpassungen und Erweiterungen

C2: ERWEITERBARKEIT In der Literatur ist gezeigt, dass die Verwendung von Modellierungssprachen oder Standards ohne weitere Anpassungen oftmals nur eingeschränkt möglich ist. Ladleif et al. haben zum Beispiel eine Semantikerweiterung für BPMN-Choreografien vorgestellt, um sie auf einer Blockchain-Infrastruktur auszuführen [96]. Weiterhin haben Köpke et al. in Bezug auf die Modellierungsphase eine Erweiterung für BPMN-Prozessdiagramme vorgestellt, um Blockchain-relevante Sicherheitsaspekte beschreiben zu können [92]. Zuletzt ist in dieser Arbeit bereits beschrieben, dass die organisationale Perspektive in BPMN nur rudimentär umgesetzt ist.

Das zu entwickelnde Artefakt soll in diesem Sinne erweiterbar gestaltet werden, sodass Änderungen oder Erweiterungen in der BPM- und SCI-Domäne, also Modellierungssprachen, Ausführungssemantiken und Blockchain-Protokolle sowie alternative Algorithmen, unkompliziert integriert, angewendet und erweitert werden können.

6.4 DPEX-FRAMEWORK: DESIGN UND ENTWICKLUNG

Dieses Kapitel stellt das Artefakt vor, welches in diesem DSR-Zyklus entwickelt wird und die Anforderungen aus Kapitel 6.3 umsetzen soll. Zuvor soll dieses Artefakt näher klassifiziert werden.

Klassifizierung des DSR-Artefakts

Wie in Kapitel 1.3.1 beschrieben, kann ein Artefakt als Ergebnis eines DSR-Projekts im Allgemeinen unterschiedlicher Natur sein. Neben einem Prototyp (*instantiation*) gelten auch ein *construct* (deutsch: Konstrukt, Konzept oder Konzeptionalisierung), ein *model* (deutsch: Modell) oder eine *method* (deutsch: Methode) als Art von Artefakt [70]. Ein *construct* ist etwa die Idee des relationalen Modells in der Datenbanktheorie, während ein *model* demnach die Umsetzung dieser Idee als Modellierungssprache darstellt, zum Beispiel in Form von Entity-Relationship-Modellen. Weiterhin umfasst eine *method* eine Menge an Schritten, einen Algorithmus oder Richtlinien, um eine bestimmte Aufgabe zu lösen, während eine *instantiation* eine Realisierung eines Artefakts in seiner IT-Umgebung begleitet, um ein spezifisches Problem zu lösen. Im Rahmen dieses DSR-Zyklus wird ähnlich einer *instantiation* auch ein Softwareartefakt erzeugt, jedoch ist das zentrale entwickelte Artefakt die konzeptionelle Idee einer Middleware-basierten Schichtenarchitektur zur Anwendungsentwicklung für die dezentrale Prozessausführung. Die Implementierung dient ausschließlich der Demonstration der Umsetzbarkeit der konzeptionellen Idee. Diese Differenzierung ist entscheidend, da im Evaluationsschritt nicht das entwickelte Softwareartefakt beurteilt wird, sondern die konzeptionelle Strategie.

Es folgt ein Gesamtüberblick über das dpex-Framework in Kapitel 6.4.1. Darauf aufbauend stellt Kapitel 6.4.2 die drei Schichten des Frameworks, BPM, SCI und Middleware vor, bevor Kapitel 6.4.3 die Architektur im Detail beschreibt. Kapitel 6.4.4 stellt die Schnittstellen zwischen den Schichten vor, während Kapitel 6.4.5 zuletzt die Interaktionen zwischen den drei Schichten erörtert.

6.4.1 *dpex-Framework: Gesamtüberblick*

Dieses Kapitel gibt einen groben Überblick über den Aufbau und die Funktionsweise von dpex. Abbildung 33 visualisiert die grundsätzliche Funktionsweise von dpex.

globales Prozessmodell

Nach dem Decentralized Control-Prinzip erfordert dpex ein Prozessmodell, welches den globalen zwischenbetrieblichen Prozess beschreibt, und damit alle Regeln für die Kollaboration kapselt (blau, links in Abbildung 33). Das Prozessmodell spezifiziert wie in Kapitel 2.5 beschrieben die zu erledigenden Aufgaben. Die Definition von Ausführungsrechten in der organisationalen Perspektive erfolgt wie üblich über die Referenzierung von Entitäten. Hierfür wird entspre-

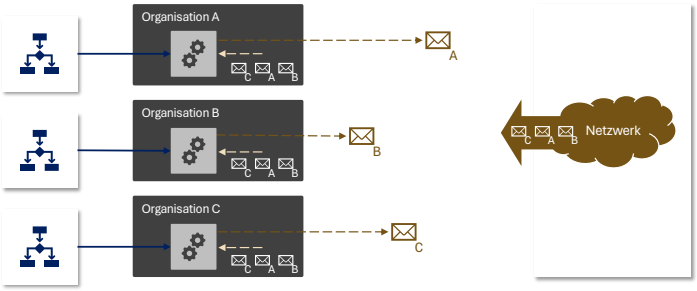

Abbildung 33: Drei Organisationen A, B und C interpretieren lokal das gleiche Prozessmodell. Die Kommunikation und Synchronisierung aller Ereignisse erfolgt über die Netzwerkkomponente.

chend ein globales, zwischenbetriebliches Organisationsmodell benötigt.

Kommunikation von Ereignissen

Während der Prozessausführung wird jede teilnehmende Organisation dieses globale Prozessmodell lokal interpretieren, zum Beispiel in einem WFMS (grau, mittig in Abbildung 33). Jede Aktion, welche den Prozessstatus aktualisiert, hat ihren Ursprung in einer bestimmten Organisation. Die Aktionen, welche beispielsweise den Lebenszyklusstatus von Aktivitäten aktualisieren, müssen über Nachrichten an alle teilnehmenden Organisationen weitergeleitet werden, damit diese den jeweils lokal verwalteten Prozessstatus ebenfalls entsprechend aktualisieren können (braune Nachrichtensymbole $\boxtimes_A$, $\boxtimes_B$, $\boxtimes_C$ in Abbildung 33).

Synchronisierung von Ereignissen

Üblicherweise können Aufgaben von mehreren Personen ausgeführt werden, welche hier möglicherweise auch in verschiedenen Organisationen beschäftigt sind. Durch die verteilte, asynchrone Infrastruktur können dabei Wettlaufsituationen entstehen, nämlich dann, wenn zum Beispiel zwei Personen annähernd gleichzeitig die Ausführungsrechte für eine bestimmte Aufgabe beanspruchen. Diese beiden Nachrichten könnten in unterschiedlicher Reihenfolge bei den teilnehmenden Organisationen eintreffen, wodurch diese auch in unterschiedlicher Reihenfolge interpretiert werden, was schließlich zu einer unterschiedlichen Auffassung des aktuellen, globalen Prozessstatus führen wird. Aus diesem Grund ist eine Synchronisierung aller Ereignisse erforderlich, so wie es auch in Blockchain-basierten Systemen erfolgt. Die Netzwerkkomponente sorgt also durch Synchronisierungsalgorithmen dafür, dass sich alle Teilnehmer auf die gleiche Reihenfolge der Nachrichten einigen. In Abbildung 33 senden die drei Teilnehmer jeweils eine Nachricht ($\boxtimes_A$, $\boxtimes_B$ und $\boxtimes_C$) an die Netzwerkkomponente. Die Netzwerkkomponente erstellt daraufhin eine global akzeptierte Reihenfolge der Nachrichten ($\boxtimes_C, \boxtimes_A, \boxtimes_B$ in Abbildung 33). Über diese Reihenfolge werden dann alle Teilnehmer benachrichtigt, welche die Nachrichten so in der selben Reihenfolge abarbeiten. Damit können alle Organisationen lokal den gleichen

Prozessstatus ableiten und so den zwischenbetrieblichen Prozess dezentral ausführen.

In den nächsten Kapiteln wird gezeigt, wie die dpex-Instanzen in den Organisationen die Benutzeranfragen verarbeiten, an die entsprechenden lokalen BPM- und SCI-Komponenten weiterleiten und externe Systeme für die Kommunikation und die Verarbeitung der Ereignisse anbinden können.

6.4.2 *Drei-Schichten-Modell von dpex*

Trennung von BPM- und SCI-Schicht

Bei Decentralized Control-Ansätzen müssen Herausforderungen aus zwei unterschiedlichen Domänen berücksichtigt werden. Folglich sind auch beim dpex-Framework die beiden Aufgabenbereiche des Prozessmanagements auf der einen Seite und die Kommunikation, Synchronisation und Sicherheit von Nachrichten auf der anderen Seite zu betrachten. Im Gegensatz zur Blockchain-basierten Prozessausführung, bei der durch die direkte Implementierung der Prozessausführung in Smart Contracts diese beiden Aspekte verwoben werden, strebt das dpex-Framework eine strikte Trennung dieser Disziplinen an. Die BPM-Schicht kapselt die Implementierung der Prozessausführung, während die SCI-Schicht für die Kommunikation, Synchronisierung und Sicherheit des Nachrichtenaustausches zuständig ist. Die Integration der beiden Schichten in die Gesamtarchitektur erfolgt durch die Middleware als Koordinationsschicht.

Middleware als Koordinationsschicht

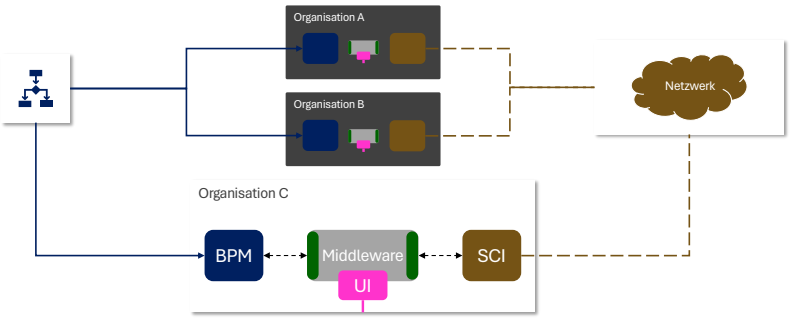

Abbildung 34: Verknüpfung der BPM-Schicht (blau) und der SCI-Schicht (braun) über eine Middleware-Schicht (grau) im dpex-Framework

Insgesamt wird das dpex-Framework also über die folgenden drei Schichten definiert (Abbildung 34).

BPM-Schicht

PROZESSMANAGEMENT (BPM) Die Prozessmanagementschicht umfasst die Beschreibung und Ausführung der Prozesse aus Sicht des Prozessmanagements, so wie in Kapitel 2 beschrieben.

SCI-Schicht

SICHERE KOMMUNIKATIONSINFRASTRUKTUR (SCI) Die SCI-Schicht kapselt die Aufgaben einer sicheren Kommunikationsinfrastruktur, welche bei dezentral verwalteten Systemen von zentraler Be-

deutung ist. Blockchain-Netzwerke können als SCI angebunden werden, jedoch existieren neben den Blockchain-Protokollen noch weitere Möglichkeiten für die Implementierung einer SCI.

Koordinationsschicht

KOORDINATIONSMIDDLEWARE Die dritte Schicht ist für die Koordination zwischen der BPM- und SCI-Schicht zuständig. Des Weiteren stellt die Middleware eine Benutzerschnittstelle für Endanwender zur Verfügung, um deren Aktionen zu organisieren und die jeweils erforderlichen (Teil-)Komponenten in der Gesamtarchitektur zu benachrichtigen und aufzurufen.

Einigung auf BPM- und SCI-Artefakte

Auf zwischenbetrieblicher Ebene müssen die Organisationen eine Einigung bezüglich zweier Artefakte finden. Zum einen beinhaltet das globale, zwischenbetriebliche Prozessmodell alle Regeln, nach denen die Kollaboration ablaufen soll. Zum anderen definiert die SCI-Schicht das Protokoll, welches die sichere Nachrichtensynchronisierung abwickeln soll. Die Integration dieser Artefakte in die Prozessausführung erfolgt jeweils über die interne Implementierung der drei Schichten in den teilnehmenden Organisationen. Die Details der einzelnen Schichten, deren Aufbau sowie deren organisationsinterne und -externe Interaktionen zur Implementierung der dezentralen Prozessausführung werden in den folgenden Kapiteln genauer dargestellt. Dabei ist die Schicht der Koordinationsmiddleware von besonderem Interesse, da die Implementierung der BPM- und SCI-Schicht bereits in Kapitel 2 anhand von BPMN und Camunda beziehungsweise in Kapitel 4 anhand der Ethereum-Blockchain diskutiert ist.

6.4.3 *Aufbau und Architektur des dpex-Frameworks*

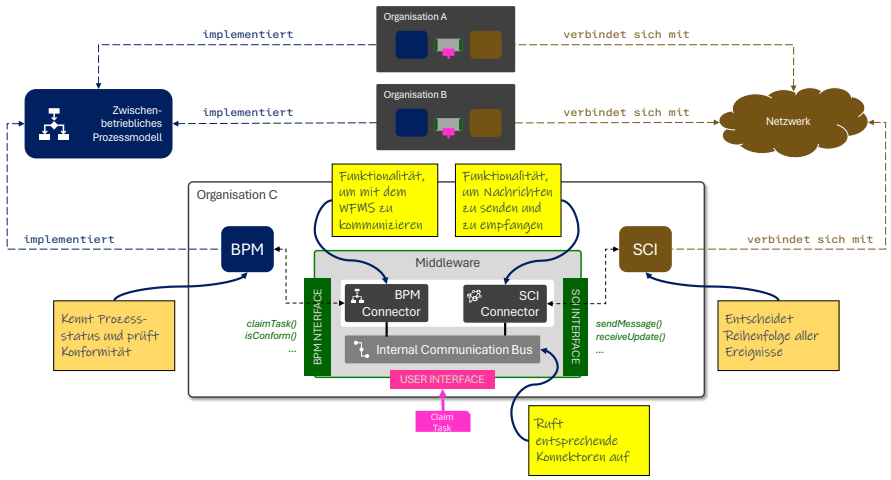

Abbildung 35: Detaillierte Architektur des dpex-Frameworks

In diesem Kapitel werden die Komponenten von dpex vorgestellt und ihre Funktionalität sowie die Integration in die Gesamtarchitek-

tur beschrieben. Abbildung 35 gibt einen Überblick über die detaillierte Architektur inklusive den Komponenten mit ihren Hauptaufgaben.

BPM-Komponente

Die BPM-Komponente kapselt die Funktionalitäten zur Prozessausführung (Kapitel 2.3) und wird in Kapitel 6.4.3.1 näher beschrieben.

SCI-Komponente

Die SCI-Komponente kümmert sich um die Kommunikation und Synchronisierung von Nachrichten unter Berücksichtigung des Sicherheitsaspekts und wird in Kapitel 6.4.3.2 näher beschrieben.

Middleware

Die Middleware ist die zentrale Komponente, da sie die externen Komponenten zur Prozessausführung und sicheren Kommunikationssynchronisierung koordiniert. Innerhalb der Middleware stehen dafür sogenannte Konnektoren zur Verfügung, welche die externen Systeme über die Implementierung von Schnittstellen (grün in Abbildung 35) anbindet. Der sogenannte Internal Communication Bus (ICB) (deutsch: interner Kommunikationskanal) koordiniert dabei die entsprechenden Konnektoren innerhalb der Middleware. Die Komponenten der Middleware werden in Kapitel 6.4.3.3 genauer vorgestellt.

ICB

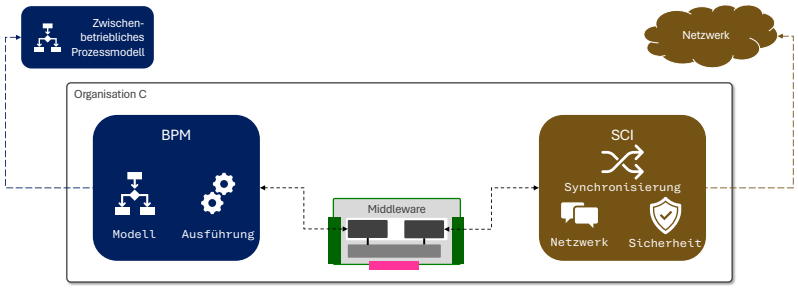

Abbildung 36: dpex-Module in der BPM- und SCI-Schicht

Module der externen Komponenten

Damit externe BPM- und SCI-Komponenten über die Schnittstellen angebunden werden können, erwartet dpex die in Abbildung 36 dargestellten Module, welche in den folgenden Kapiteln erläutert werden. Die Module können in der Middleware in den entsprechenden Konnektoren auch erweitert oder manuell implementiert werden (Kapitel 6.5).

6.4.3.1 *BPM-Komponente*

Die BPM-Komponente verwaltet zwei verschiedene Module, welche sich auf die Phasen der Modellierung und der Implementierung beziehungsweise der Ausführung beziehen (links in Abbildung 36). Auf der einen Seite dient das Prozessmodellartefakt zur Definition von Prozessmodellen, welche über dpex ausgeführt werden sollen. Da dpex keine Synchronisierung oder öffentliche Repositorien von Prozessmodellen bietet, muss jeder Teilnehmer in der lokalen dpex-Instanz jeweils das zwischenbetriebliche Prozessmodell als Artefakt definieren. Das Prozessmodellartefakt beinhaltet dabei neben dem Prozess auch alle weiteren zur Ausführung notwendigen und global einheit-

Prozessmodell-artefakt

lichen Strukturen und Informationen wie das Organisationsmodell oder auszuführende Skripte.

Ausführungsartefakt

Auf der anderen Seite definiert die BPM-Komponente die Ausführungsartefakte, welche die spezifizierten Prozessmodelle nach Kapitel 2.3 interpretieren können. Dies umfasst, wie in Kapitel 2.4 beschrieben, unter anderem die Instanziierung von Prozessmodellen, die Verwaltung des Prozessstatus pro Instanz in einem Ereignisprotokoll, die Prüfung auf Konformität bezüglich eines Prozessschritts oder die Bereitstellung von Arbeitslisten. Die Artefakte sind wiederverwendbar, das bedeutet, dass ein Ausführungsartefakt, etwa ein bestimmtes WFMS, für die Interpretation von mehreren Prozessmodellen auch unterschiedlicher Kollaborationen verwendet werden kann. Eine Wiederverwendung von Prozessmodellen ist im allgemeinen Fall nicht sinnvoll, da unter anderem kollaborationsspezifische Informationen im Organisationsmodell integriert sind.

Verwendung externer Systeme

Grundsätzlich übernimmt die BPM-Komponente damit genau die Kernaufgaben, welche von gängigen Prozessausführungssystemen für die Ausführung von traditionellen, innerbetrieblichen Prozessen übernommen werden. Mit dpex können diese Funktionen manuell implementiert werden, um kollaborationsspezifische Anforderungen umzusetzen. Im besten Fall hingegen einigen sich die Kollaborationspartner auf ein Prozessmodell, welches mit gängigen WFMSs ausgeführt werden kann. Dies erlaubt die Integration bestehender, externer Systeme für die Prozessausführung in dpex. Weiterhin ist es in dpex möglich, die Funktionalität externer Systeme zu erweitern. Dies wird in Kapitel 6.5 im Rahmen einer Erweiterung des Camunda WFMS in Bezug auf eine fortgeschrittene Unterstützung der organisationalen Perspektive demonstriert.

6.4.3.2 *SCI-Komponente*

Die SCI-Komponente sorgt für die Kommunikation von Nachrichten, deren Synchronisierung sowie für die Umsetzung bestimmter Sicherheitsaspekte (rechts in Abbildung 36). Ähnlich zur BPM-Komponente können entweder externe Systeme und Algorithmen eingebunden werden oder deren Aufgaben und Funktionen manuell über das dpex-Framework implementiert werden. Die Kernaufgaben der SCI-Komponenten können anhand Blockchain-basierter Systeme (Kapitel 4) abgeleitet werden, da diese sich für die Integration als externes System in dpex eignen.

Für deren Anbindung oder eine manuelle Implementierung der SCI-Schicht definiert das dpex-Framework die Module Netzwerk, Sicherheit und Synchronisierung, welche sich im Grunde aus den wichtigsten Eigenschaften Blockchain-basierter Systeme für die dezentrale Prozessausführung ableiten und folgendermaßen in das dpex-Framework integriert werden.

Kommunikation aller Ereignisse

NETZWERK- UND KOMMUNIKATIONSMODUL Die Kommunikation von Informationen und die Verteilung aller Nachrichten beziehungsweise Ereignisse ist von zentraler Bedeutung bei dezentralen Systemen. Bei dpex erfolgt die Verwaltung des Prozessstatus durch die lokale Interpretation der Modelle in den jeweiligen BPM-Komponenten ebenfalls dezentral, womit die Kommunikation als essenzieller Baustein im Framework betrachtet werden kann, um die Datenstände synchron zu halten. Das Netzwerkmodul muss also die Funktionen `send` zum Senden von Nachrichten und `receive` zum Erhalten von Nachrichten implementieren. Bei manueller Implementierung kann das Senden über die IP-Adressen der Teilnehmer und das HTTP-Protokoll abgewickelt werden. Wenn ein Blockchain-System als SCI-Komponente angebunden wird, wickelt der entsprechende Konnektor der dpex-Instanz das Senden der Nachricht als Transaktion über den externen Blockchain-Knoten ab, sodass die Transaktion automatisch über das Blockchain-Protokoll an alle Teilnehmer übermittelt wird. Für das Empfangen von Nachrichten kann sowohl eine *push*-Strategie als auch eine *pull*-Strategie verfolgt werden. Bei der *pull*-Strategie überprüft der Netzwerkadapter in dpex proaktiv einen externen Dienst, ob neue Nachrichten verfügbar sind. Bei der *push*-Strategie wird der Netzwerkadapter von einem externen Dienst aufgerufen und reaktiv benachrichtigt. Die *push*-Strategie findet zum Beispiel bei der Anbindung der Ethereum-Blockchain als SCI-Komponente Anwendung und wird dabei durch das Event-System der EVM und entsprechende Event-Handler im Netzwerkadapter umgesetzt (Kapitel 6.5.3.2).[4]

send und receive

push versus pull beim Empfang von Nachrichten

keine vollständige Spezifikation

SICHERHEITSMODUL Die Sicherheit von Softwaresystemen ist ein komplexes Thema und es ist hier nicht Ziel des Sicherheitsmoduls, die vollständigen Sicherheitsanforderungen an eine SCI für dpex zu definieren oder abzubilden. Ein Grund dafür ist, dass weitere Sicherheitsaspekte wie die Robustheit gegen Manipulationsversuche vom Synchronisierungsmodul auch von anderen Modulen umgesetzt werden. Das im dpex-Framework definierte Sicherheitsmodul der SCI-Komponente ist vordergründig für die Authentifizierung von Nutzern zuständig. Damit trägt es einerseits zum Sicherheitsaspekt der Zugriffskontrolle bei und beugt andererseits dem Angriffsvektor der Repudiation vor, also der Möglichkeit für Nutzer, die tatsächlich getätigten Aktionen im Nachhinein zu leugnen. Anhand von Blockchain-basierten Systemen ist gezeigt, dass *Non-Repudiation* eine entscheidende Sicherheitskomponente derartiger dezentraler Systeme ist. Mithilfe der Methoden der asymmetrischen Kryptographie und insbesondere digitalen Signaturen (Kapitel 4.2.2) kann die Implementierung auch direkt in dpex umgesetzt werden. Über das Sicherheitsmodul in

Authentifizierung

sign und verify

4 Aus technischer Sichtweise, kann die Überwachung der Transaktionslogs im Blockchain-Speicherbereich auch als pull-Strategie betrachtet werden.

dpex können damit die `sign`-Funktion für das Signieren von Nachrichten und die `verify`-Funktion für das Verifizieren einer Signatur implementiert werden.

SYNCHRONISIERUNGSMODUL Bei verteilten Systemen besteht die grundsätzliche Herausforderung, eine global eindeutige Reihenfolge aller auftretenden Nachrichten oder Ereignisse über alle Teilnehmer hinweg abzuleiten [97]. Durch die jeweils lokale Interpretation der Prozessmodelle und damit einhergehende dezentrale Speicherung der Daten zusammen mit der asynchronen Kommunikationsstrategie ist in dpex ebenfalls eine Synchronisierung aller Ereignisse obligatorisch. Die Umsetzung in dpex erfolgt über das Synchronisierungsmodul. Das Netzwerkmodul empfängt eingehende Nachrichten, ohne dass diese Nachrichten oder Ereignisse bereits eine Auswirkung auf den Prozessstatus haben. Die Ereignisse werden im Synchronisierungsmodul verwaltet und mittels eines geeigneten Algorithmus überprüft, ob das Ereignis als *final* angesehen werden kann, oder ob diese Nachricht in der globalen Ordnung noch durch eine andere Nachricht revidiert werden kann. Erst wenn das Synchronisierungsmodul die Dauerhaftigkeit einer einkommenden Nachricht bestätigt, wird das dpex-Framework das entsprechende Ereignis zur internen Verarbeitung weiterleiten.

global eindeutige Reihenfolge

Nachrichtenempfang ohne Auswirkung

isFinal

Interpretation nach Feststellung der dauerhaften Persistenz

Aufgrund der Komplexität, BFT-resiliente Systeme zu implementieren, wird mit dem dpex-Framework die Anbindung externer Systeme wie Blockchains oder BFT-SMaRt [12] einer Neuimplementierung der Synchronisierung vorgezogen. Ähnlich zur BPM-Komponente können diese externen Systeme in den Konnektoren der SCI-Schicht erweitert werden, wie in Kapitel 6.6 am Beispiel der probabilistischen Finalität diskutiert.

6.4.3.3 *dpex-Middleware*

Die dpex-Middleware ist das Kontrollzentrum des Frameworks und setzt sich aus den drei Komponenten BPM-Konnektor, SCI-Konnektor und ICB zusammen. Für jedes externe System muss dabei jeweils ein BPM-Konnektor und ein SCI-Konnektor implementiert werden, zum Beispiel ein `CamundaAdapter` oder ein `EthereumAdapter`. Für jede Kollaboration wird jeweils ein passender Konnektor ausgewählt, instanziiert und entsprechend konfiguriert. Zum Beispiel wird der `CamundaAdapter` mit der entsprechenden URL des zu verwendenden externen Camunda WFMS konfiguriert, während der `EthereumAdapter` mit der URL des lokal laufenden Ethereum-Knotens konfiguriert wird. Die BPM-Schnittstelle und die SCI-Schnittstelle stellen dabei die Interoperabilität zwischen den Konnektoren der Middleware und den externen Komponenten und deren Modulen sicher. Die Schnittstellen fordern die Implementierung bestimmter Funktionen in den Konnektoren, zum Beispiel `claimTask` oder `completeTask` beziehungsweise

Aufbau der Middleware

`sendMessage` oder `receiveMessage`. In diesen Funktionen wird in den Konnektoren die Kommunikation mit den extern angebundenen Systemen implementiert. So ruft die `completeTask`-Methode in einem `CamundaAdapter` den entsprechenden REST-API-Endpunkt des konfigurierten externen Camunda WFMS für das Abschließen einer Aktivität auf.

Aufruf der Middleware

Die dpex-Middleware wird während der Prozessausführung entweder über die Benutzerschnittstelle oder von der SCI-Komponente über externe Ereignisse aufgerufen. Der ICB ist dabei dafür zuständig, sowohl die Ereignisse der Benutzerschnittstelle als auch die externen Ereignisse zu verarbeiten und entsprechend weiterzuleiten. Zum Beispiel wird durch die vorherige Definition eines Event-Handlers der SCI-Konnektor durch Ereignisse des externen Ethereum-Netzwerks aufgerufen. Der Konnektor übergibt das Ereignis und den Kontrollfluss dann an den ICB. Der ICB interpretiert das Ereignis und ruft den entsprechenden BPM-Konnektor der Kollaboration auf. Dieser sorgt schließlich dafür, dass der lokal verwaltete Prozessstatus im externen WFMS aktualisiert wird.

Konnektoren als Mediator zwischen ICB und externen Komponenten

In der Gesamtarchitektur dienen die Konnektoren also als Mediator zwischen der dpex-Middleware und den externen Komponenten und leiten den Kontrollfluss entsprechend weiter.

6.4.3.4 *Kurze Diskussion der Architektur*

Kapitel 6.4.3.1, Kapitel 6.4.3.2 und Kapitel 6.4.3.3 definieren die drei Schichten des dpex-Frameworks. Insbesondere werden innerhalb der BPM- und SCI-Schicht Module definiert, welche entweder von den externen Komponenten bereitgestellt werden oder manuell implementiert werden müssen. Wenn ein extern angebundenes System der BPM- oder SCI-Schicht eine Funktion nicht zur Verfügung stellt, so kann beziehungsweise muss diese Funktionalität manuell implementiert werden (Kapitel 6.5). Weiterhin stellt sich die Frage, ob die Schichten ausreichend oder vollständig beschreiben sind. Aus Anwendersicht ist diese Frage im Kontext dieses DSR-Zyklus allerdings nicht zu beantworten. Für das DSR-Projekt formuliert Kapitel 6.3 die Anforderungen, welche vom Artefakt zu erfüllen sind. Dies wird in Kapitel 6.6 basierend auf der Demonstration des Artefakts in Kapitel 6.5 evaluiert. In diesem Sinne wird gezeigt, dass mit den in dpex definierten Schichten beziehungsweise Modulen eine Anwendung zur zwischenbetrieblichen Prozessausführung realisiert werden kann. Diese Anwendung erlaubt es dabei die in Kapitel 6.3 postulierten Anforderungen umzusetzen. In zukünftigen Studien kann basierend auf dieser konzeptionellen Arbeit gezeigt werden, inwieweit das hier vorgestellte DSR-Artefakt in der Praxis eingesetzt werden kann. In den anwendungsbezogenen Feldstudien können weitere Anforderungen identifiziert werden, welche einen nächsten Zyklus zur Verbesserung des dpex-Frameworks veranlassen. Es ist jedoch nicht das Ziel eines

DSR-Projekts, die Anforderungsdefinition auf Vollständigkeit zu evaluieren.

6.4.4 *Schnittstellen zwischen den Schichten*

Interoperabilität

Um trotz der strikten Entkopplung der BPM- und SCI-Schicht die Interoperabilität in dpex zu gewährleisten, definiert das Framework drei Schnittstellen, nämlich die BPM-Schnittstelle, die SCI-Schnittstelle und die Benutzerschnittstelle (grün beziehungsweise pink in Abbildung 35). Die BPM-Schnittstelle umfasst Funktionen, welche zur Ausführung von Prozessen notwendig sind, zum Beispiel `claimTask` oder `completeTask` zum Aktualisieren des Lebenszyklusstatus einer Aktivität. Die SCI-Schnittstelle spezifiziert Funktionen, um die einzelnen Module an dpex anzubinden, zum Beispiel `send` und `receive` für das Netzwerkmodul, `sign` und `verify` für das Sicherheitsmodul oder `isAgreementReached` für das Synchronisierungsmodul.

BPM-Schnittstelle

SCI-Schnittstelle

Der genaue Funktionsumfang der Schnittstellen ist durch das Framework nicht fest vorgegeben. Stattdessen können Implementierungen des Frameworks unterschiedliche Strategien verfolgen. Dies soll an einem Beispiel gezeigt werden.

Flexibilität in den Schnittstellen

Während Funktionen der informationsorientierten Perspektive, zum Beispiel `updateData`, explizit in der BPM-Schnittstelle definiert sein könnten, werden in der Implementierung in Kapitel 6.5 Prozessvariablen über einen Parameter der `completeTask`-Funktion beim Abschließen einer Aufgabe aktualisiert. Des Weiteren könnte die BPM-Schnittstelle um Funktionen `cancelTask` (deutsch: Aufgabe abbrechen) oder `reassignTask` (deutsch: Aufgabe neu zuweisen) erweitert werden, um den Aktivitätslebenszyklus detaillierter abzubilden. Alternativ kann die Schnittstelle auch flexibler gestaltet werden, indem lediglich eine Funktion `updateActivityStage` definiert wird, welche den Lebenszyklusstatus als Parameter übergeben bekommt.

6.4.5 *Kommunikationsschema*

Der komplette Programmfluss wird in dpex über den ICB gesteuert. Der ICB stellt über die Konnektoren das Bindeglied zwischen den externen BPM- und SCI-Komponenten sowie den Endanwendern dar und wird entweder durch eine Benutzeraktion oder durch ein Ereignis der externen SCI-Komponente aufgerufen. Dieses Kapitel beschreibt die Kommunikationsschemata innerhalb des dpex-Frameworks und die interne Koordination des Programmflusses. Grundsätzlich unterteilt sich der Programmablauf des dpex-Frameworks in das Senden einer Aktion oder einer Nachricht auf der einen Seite und das Empfangen einer Nachricht auf der anderen Seite.

Senden einer Nachricht

SENDEN EINER NACHRICHT Der Programmablauf für das Senden einer Nachricht in dpex ist in Abbildung 37 schematisch dargestellt. Die einzelnen Schritte werden in der folgenden Auflistung näher beschrieben. Die Nummerierung referenziert dabei die Ziffern in Abbildung 37.

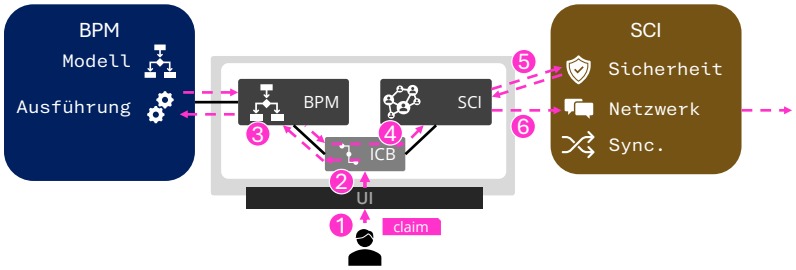

Abbildung 37: Ablauf für das Senden einer Nachricht in dpex

1. Der Endanwender führt eine Aktion über die Benutzerschnittstelle aus.
2. Die Aktion wird verarbeitet und zusammen mit dem Kontrollfluss an den ICB weitergeleitet.
3. Der BPM-Konnektor wird aufgerufen, welcher über die externe BPM-Komponente die Aktion prüft, um nicht konforme Aktionen im Netzwerk zu verhindern.
4. Die Aktion wird als Nachrichtenobjekt an den SCI-Konnektor weitergeleitet.
5. Das Sicherheitsmodul der externen SCI-Komponente signiert die Nachricht.
6. Das Netzwerkmodul der externen SCI-Komponente verschickt die Nachricht.

Empfangen einer Nachricht

EMPFANGEN EINER NACHRICHT Nachdem eine Nachricht versendet ist, endet der Programmablauf vorerst, da möglicherweise eine konkurrierende Nachricht von einem weiteren Teilnehmer versendet wurde. Das heißt, bevor sich die Benutzeraktion auf den lokalen Datenstand auswirkt, muss diese zuerst an alle Teilnehmer versendet werden. Daraufhin kann über die SCI eine global eindeutige, nicht revidierbare Ordnung aller Transaktionen hergestellt werden. Erst dann werden die lokalen Datenstände aller Teilnehmer aktualisiert. Dieses Vorgehen ist im Kommunikationsschema für das Empfangen einer Nachricht in Abbildung 38 dargestellt. Die folgende Auflistung beschreibt dabei wieder den Programmablauf und referenziert die Ziffern in der Abbildung.

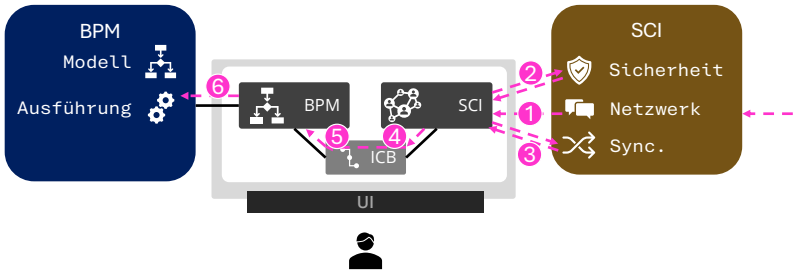

Abbildung 38: Ablauf für das Empfangen und Verarbeiten einer Nachricht in dpex

1. Das Netzwerkmodul aktiviert den SCI-Konnektor durch die Weiterleitung einer eingehenden Nachricht. Alternativ überprüft der Konnektor aktiv wartende, eingehende Nachrichten.

2. Das Sicherheitsmodul überprüft die Signatur der eingehenden Nachricht.

3. Das Synchronisierungsmodul entscheidet, ob die Nachricht weiterverarbeitet wird oder ob die Nachricht bisher nicht synchronisiert ist. Falls die Prüfung nicht erfolgreich ist, endet der Programmfluss hier und es muss gegebenenfalls auf weitere Nachrichten gewartet werden (Kapitel 6.5.3.6).

4. Bei einer erfolgreichen Prüfung wird die Nachricht an den ICB weitergeleitet.

5. Der ICB identifiziert den aufzurufenden BPM-Konnektor und leitet die Nachricht als Aktion weiter.

6. Der BPM-Konnektor ruft die entsprechende Funktionalität der externen BPM-Komponente auf, um die Aktion zu verarbeiten und den lokalen Datenstand zu aktualisieren.

6.5 DEMONSTRATION: DPEX-FRAMEWORK IMPLEMENTIERUNG

Dieses Kapitel präsentiert eine mögliche Implementierung des dpex-Frameworks, um dessen Umsetzbarkeit zu belegen. Es wird gezeigt, wie die Middleware realisiert wird und gleichzeitig wird illustriert, wie externe BPM- und SCI-Komponenten über Konnektoren integriert und erweitert werden können.

Im Folgenden stellt Kapitel 6.5.1 die verschiedenen Teilprojekte der Implementierung des dpex-Frameworks vor, bevor Kapitel 6.5.2 die wichtigsten Aspekte in Bezug auf die Softwarearchitektur der Anwendung vorstellt. Zuletzt zeigt Kapitel 6.5.3 die Implementierung eines Camunda- und Ethereum-Adapters.

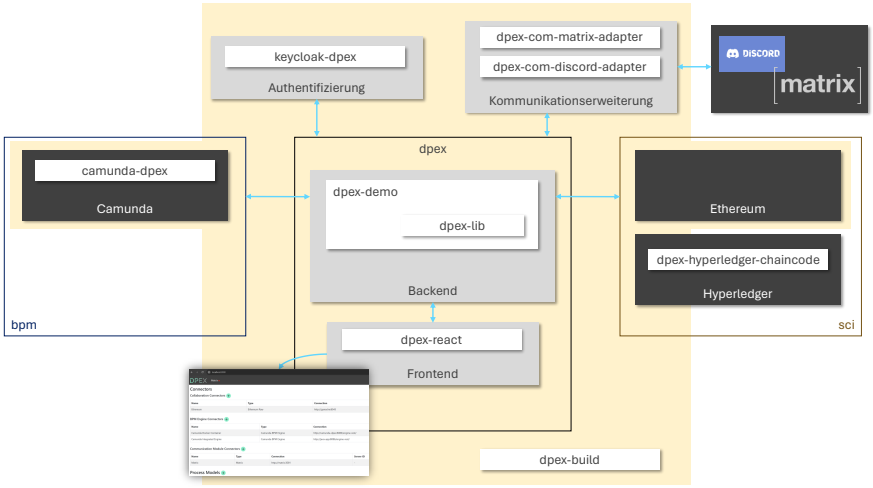

Abbildung 39: Struktur der Gitlab-Repositorien

6.5.1 *Projektstrukturierung und Quelltext-Verwaltung*

Die Implementierung von dpex ist auf mehrere Git-Projekte verteilt und wird in verschiedenen Quelltext-Repositorien verwaltet, welche auf GitLab[5] öffentlich zugänglich sind und hier auszugsweise kurz aufgelistet werden. Aufgrund der stetigen Weiterentwicklung werden immer wieder weitere Repositorien hinzugefügt. Weitere Informationen zur Verwendung finden sich in den jeweiligen `README.md`-Dateien. Die Repositorien sind zusammen mit der jeweiligen Funktionalität in Abbildung 39 dargestellt.

BIBLIOTHEK: DPEX-LIB Die Koordinationsmiddleware als Kernkomponente ist in `dpex-lib` implementiert und als Programmbibliothek auf Basis von Java und des Spring Boot-Frameworks konzipiert. Damit ist `dpex-lib` selbst keine ausführbare Software, sondern als Bibliothek für die Verwendung in einer implementierenden Spring-Boot-Anwendung vorgesehen.

BACKEND: DPEX-DEMO Das `dpex-demo`-Projekt demonstriert, wie die `dpex-lib`-Bibliothek in eine ausführbare Software integriert werden kann. `dpex-demo` basiert aus Kompatibilitätsgründen ebenfalls auf Java und Spring Boot und definiert `dpex-lib` als Abhängigkeit. `dpex-demo` ist eine ausführbare Web-Anwendung (Backend), welche zur Kommunikation eine REST-API-Schnittstelle zur Verfügung stellt. In `dpex-demo` können die abstrakten Klassen aus `dpex-lib` erweitert werden, um BPM- und SCI-Adapter zu implementieren.

FRONTEND: DPEX-REACT Eine grafische Benutzerschnittstelle wird als Frontend im `dpex-react`-Projekt bereitgestellt, welches auf dem

5 https://gitlab.com/bpm-dpex/, besucht am 21.04.2024

JavaScript-Framework React basiert. Mit `dpex-react` können die Endanwender über eine grafische Benutzerschnittstelle mit dem `dpex-demo`-Backend interagieren.

DPEX-HYPERLEDGER-CHAINCODE Für die Verwendung des SCI-Adapters zur Anbindung der Hyperledger-Blockchain muss auf dieser ein Smart Contract (genannt *Chaincode*) bereitgestellt werden. Die Verwaltung der externen Hyperledger-Blockchain sowie deren Initialisierung mit dem Smart Contract liegt nicht im Verantwortungsbereich der dpex-Anwendung, weswegen der Chaincode in einem extra Repositorium organisiert ist.

Im Übrigen muss eine Ethereum-Blockchain in ähnlicher Weise mithilfe eines Smart Contracts für die Integration in das dpex-Framework vorbereitet werden. Allerdings ist im Vergleich zu Hyperledger eine unkomplizierte Initialisierung aus der dpex-Anwendung heraus möglich. Um dies zu unterstützen, wird der Quelltext des Smart Contracts für die Ethereum-Blockchain direkt im `dpex-lib`-Projekt verwaltet.

DPEX-COMMUNICATION-*-ADAPTER Zum Zeitpunkt des Schreibens dieser Dissertation wird eine Erweiterung für dpex entwickelt, mit dem Ziel, eine pseudonymisierte Kommunikation zwischen den Akteuren der Prozessausführung über Unternehmensgrenzen hinweg zu ermöglichen [41]. Für die Umsetzung werden das Matrix-Protokoll[6], ein offenes Netzwerk für sichere, dezentrale Kommunikation, und der Discord-Dienst[7] evaluiert. Die Implementierung erfolgt dann über den `dpex-communication-matrix-adapter` beziehungsweise über den `dpex-communication-discord-adapter`, deren Quelltexte wiederum in extra Repositorien verwaltet werden. Bei entsprechender Konfiguration synchronisieren diese Kommunikationsadapter den jeweiligen Nachrichtenaustausch dann mit der dpex-Anwendung.

KEYCLOAK-DPEX Ähnlich zu den Kommunikationsadaptern befindet sich aktuell auch ein Authentifizierungssystem für dpex in der Entwicklung [81]. Dieses wird im `keycloak-dpex`-Repositorium verwaltet.

CAMUNDA-DPEX Das `camunda-dpex`-Repositorium stellt ein Camunda WFMS zur Verfügung. Diese Camunda-Instanz erlaubt durch integrierte Plug-Ins in der grafischen Benutzerschnittstelle eine direkte Interaktion mit dem `dpex-demo`-Backend [191]. Dies ist als Alternative zum `dpex-react`-Frontend zu sehen.

DPEX-BUILD Die oben vorgestellten Projekte des dpex-Ökosystems werden über GitLab verwaltet und stellen unter Verwendung der inte-

6 https://matrix.org/, besucht am 14.12.2023
7 https://discord.com/, besucht am 04.03.2024

grierten *GitLab Pipeline* sogenannte *Docker-Images* zur Verfügung. Diese ausführbaren Abbilddateien können über den Containerisierungsdienst *Docker*[8] bezogen und ausgeführt werden. Dafür definiert jedes Projekt, zu dem unter anderem `dpex-react`, `dpex-demo` oder die Kommunikationsdienste gehören, ein sogenanntes *Dockerfile*, worüber die automatisierte Pipeline das Image erstellen und in den entsprechenden Container-Repositorien der Projekte in GitLab zur Verfügung stellen kann. Das `dpex-build`-Projekt beinhaltet schließlich eine sogenannte *Docker-Compose*-Datei, welche spezifiziert, wie die vorhandenen Abbilddateien als Container in einem Netzwerk über die Docker-Plattform bereitgestellt und ausgeführt werden müssen.

Neben den genannten Projekten wird für eine lokale Demonstration der Anwendung außerdem *ganache*[9], eine Simulationsumgebung für die Ethereum-Blockchain, über ein zusätzliches Docker-Image integriert, welches im Container-Repositorium des `dpex-build`-Projekts verwaltet wird. Dadurch kann der Einsatz des Ethereum-Protokolls als SCI evaluiert werden, ohne dass ein echtes Ethereum-Netzwerk konfiguriert werden muss.

Starten der Anwendung

Um die im Folgenden vorgestellte Implementierung des dpex-Frameworks als Anwendung zu starten, muss das `dpex-build`-Projekt in der Version `v0.2` über *git* geklont oder direkt über diesen Link[10] heruntergeladen werden. In dem Ordner ist anschließend über ein Terminal der Befehl `docker-compose up` auszuführen, während als Voraussetzung Docker auf dem System zu installieren und zu starten ist. Die folgenden Ausführungen beziehen sich auf die Version `v0.2` von `dpex-demo` und `dpex-react`, für welche ein *Tag* im `dpex-build`-Projekt verfügbar ist.

8 https://www.docker.com/products/docker-desktop/, besucht am 17.12.2023
9 https://www.npmjs.com/package/ganache, besucht am 17.12.2023
10 https://gitlab.com/bpm-dpex/dpex-build/-/tree/v0.2, besucht am 08.03.2024

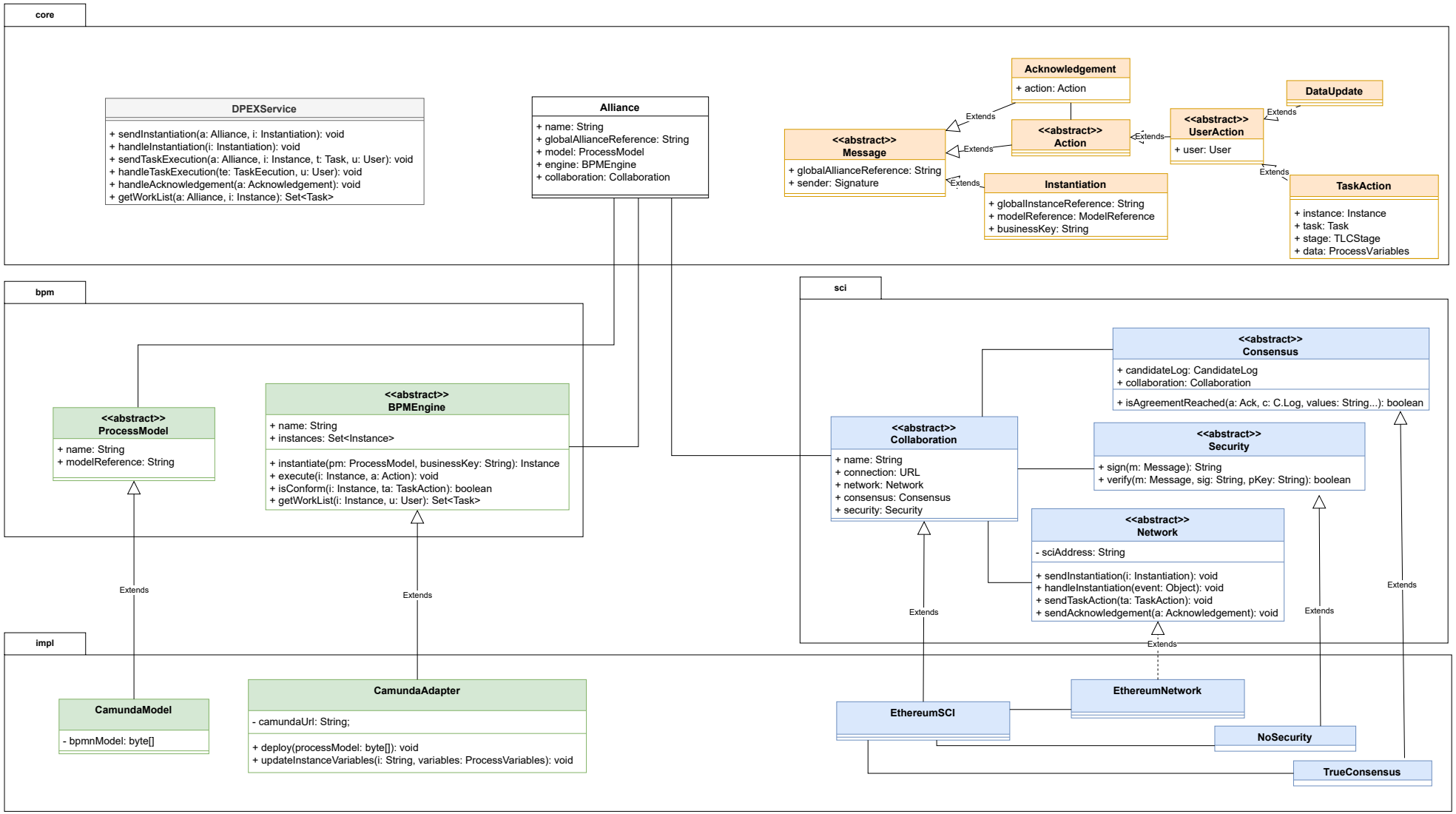

Abbildung 40: Struktur der dpex-Implementierung

6.5.2 *Architektur der dpex-Anwendung*

Die Architektur der dpex-Implementierung orientiert sich an den Komponenten der Middleware (Abbildung 35). Die Klassenstruktur der Bibliothek `dpex-lib` beziehungsweise der implementierenden Anwendungen ist in Abbildung 40 dargestellt. Die Abbildung fokussiert sich dabei auf die wesentlichen Elemente und abstrahiert von Hilfsstrukturen beispielsweise zur Datenpersistierung. Der Quelltext strukturiert sich demnach unter anderem in die Pakete `bpm`, `sci`, `core` und `impl`, welche im `dpex-lib`-Projekt verwaltet und im Folgenden erläutert werden.[11]

Das bpm-Paket

Process Model und BPMEngine

Über das `bpm`-Paket stellt die `dpex-lib` die abstrakten Klassen `ProcessModel` und `BPMEngine` zur Verfügung, welche als Korrespondenz zur BPM-Schnittstelle zu verstehen sind und somit für die Anbindung externer WFMSs zuständig sind oder die benötigte Funktionalität manuell implementieren. Insbesondere werden die Funktionen für die Instanziierung eines Prozessmodells (`instantiate`) oder zur Aktualisierung des Prozessstatus (`claim` oder `complete`) definiert. Diese abstrakten Klassen werden zur Abbindung von externen BPM-Komponenten in spezialisierten Klassen erweitert, worin die spezifizierten Methoden implementiert werden. Das `bpm`-Paket enthält weiterhin die Klassen `EventLog`, `Instance`, `ProcessVariables` und `Task`, welche die relevanten Daten der Prozessausführung kapseln.

Das sci-Paket

Das `sci`-Paket definiert die abstrakten Klassen der Koordinationsschicht. Dabei stellt `Network` die Funktionalität für die Kommunikation, also dem Senden und Empfangen von Nachrichten, zur Verfügung. `Security` definiert die Methoden zur Implementierung digitaler Signaturen (`sign` und `verify`) und kümmert sich somit um bestimmte Sicherheitsaspekte innerhalb des dpex-Frameworks. Zuletzt kann die Klasse `Consensus` im Rahmen der Synchronisierung von Nachrichten eingesetzt werden. Die drei Module werden innerhalb der Klasse `Collaboration` gekapselt. Analog zum `bpm`-Paket werden die Klassen des `sci`-Pakets erweitert, um externe Systeme der SCI-Domäne wie die Ethereum-Blockchain an dpex anzubinden.

Network

Security

Consensus

Collaboration

Das core-Paket

Im `core`-Paket befinden sich mit der Kommandozentrale `DPEXService` und `Alliance` die wichtigen Klassen zur Steuerung des Programm-

11 Die Erläuterungen können von der tatsächlichen Implementierung zugunsten einer abstrakteren, verständlicheren Darstellung abweichen.

flusses und zur Verwaltung von Kollaborationen. Der `DPEXService` fungiert als ICB und wird einerseits durch Anfragen über die Benutzerschnittstelle aufgerufen und andererseits durch das Eintreffen externer Nachrichten über eine SCI-Komponente. Je nach Art des Ereignisses leitet der `DPEXService` die nächsten Schritte ein und ruft die Funktionalität der verknüpften Konnektoren auf. Dabei spielt die `Alliance` eine zentrale Rolle. Eine `Alliance` repräsentiert eine Kollaboration beziehungsweise einen bestimmten zwischenbetrieblichen Prozess und referenziert die Konnektoren, welche für die jeweilige Kollaboration zu verwenden sind. Wenn also ein externes Ereignis über einen Event-Handler in einem `Network`-Modul empfangen wird, kann der `DPEXService` über das dazugehörige `Alliance`-Objekt den konfigurierten BPM-Konnektor identifizieren und das Ereignis korrekt weiterleiten.

DPEXService

Alliance

Das impl-Paket

Um die Verwendung der `dpex-lib` zu demonstrieren, stellt das `impl`-Paket Beispielimplementierungen der oben genannten abstrakten Klassen zur Verfügung. Diese beinhalten Adapter zur Anbindung des Camunda WFMS (`CamundaModel` und `CamundaAdapter`), zur Integration der Ethereum-Blockchain (`EthereumSci`, `Ethereum-Network`, `NoSecurity` und `TrueConsensus`)[12] oder der Hyperledger-Blockchain (`Hyperledger-Adapter` und `HyperledgerNetwork`).[13] Die Implementierungen können somit in Projekten, welche `dpex-lib` als Abhängigkeit einbinden (`dpex-demo`), verwendet werden. Die Konnektoren müssen dadurch nicht für jede Anwendung neu implementiert werden. Die Klassen werden bei der Beschreibung der Funktionsweise von dpex in Kapitel 6.5.3 im Detail vorgestellt. Die Anbindung weiterer WFMSs oder SCI-Protokolle erfolgt in der `dpex-demo`-Anwendung über die Implementierung eigener Konnektoren unter Erweiterung der jeweiligen Basisklassen.

Beispieladapter für Camunda, Ethereum und Hyperledger

6.5.3 *Die Funktionsweise der dpex-Implementierung*

Um eine Kollaboration mit dem dpex-Framework umzusetzen, müssen die Kollaborationspartner die folgenden Schritte ausführen, welche im Prozessmodell in Abbildung 41 dargestellt sind. Zu Beginn müssen die globalen Artefakte festgelegt werden, die für die zwischenbetriebliche Kollaboration mit dpex gelten sollen (Kapitel 6.5.3.1). Dies umfasst einerseits in der Modellierungsphase die kollaborative Abstimmung bezüglich eines globalen Prozess- und Organisations-

Modellierung

12 Aufgrund der Charakteristika des Ethereum-Protokolls werden mit `NoSecurity` und `TrueConsensus` Attrapenimplementierungen für die entsprechenden Module verwendet.

13 Für die Konfiguration von `HyperledgerAdapter` werden ebenfalls `NoSecurity` und `TrueConsensus` verwendet.

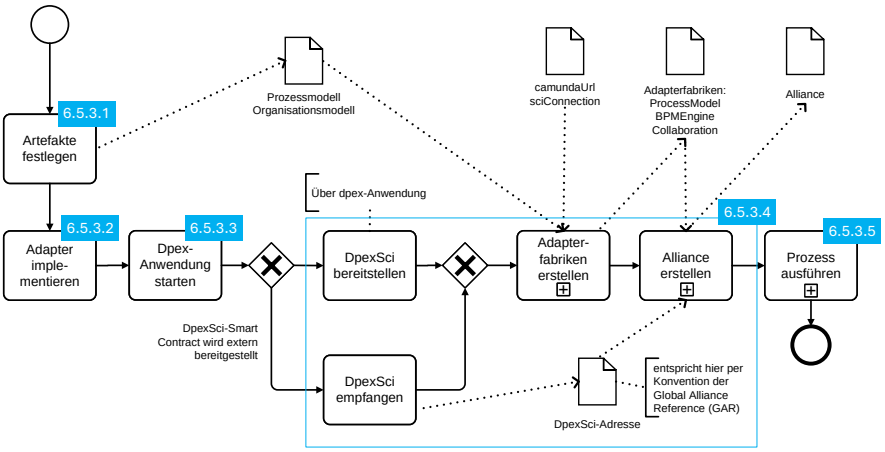

Abbildung 41: Verwendung der dpex-Anwendung

SCI bestimmen

modells. Außerdem müssen sich die Kollaborationspartner hinsichtlich einer SCI einigen, das bedeutet, der Algorithmus oder das externe System sowie dessen Konfiguration müssen festgelegt werden.

Implementierung der Konnektoren

Im Anschluss müssen die Konnektoren implementiert werden (Kapitel 6.5.3.2), welche die externen Systeme an dpex anbinden, sofern diese nicht bereits über die Beispielimplementierungen im `impl`-Paket der `dpex-lib`-Bibliothek zur Verfügung gestellt werden.

Starten von dpex

Danach muss die dpex-Anwendung gebaut und gestartet werden (Kapitel 6.5.3.3), um darin die Kollaboration entsprechend den Abmachungen mithilfe eines `Alliance`-Objekts zu konfigurieren (Kapitel 6.5.3.4).

Instanziierung Prozessausführung

Schließlich kann die Prozessausführung starten, indem eine Instanz von dem globalen Prozessmodell erzeugt wird (Kapitel 6.5.3.5). Den Aktivitätslebenszyklus folgend kann dann ein berechtigter Prozessteilnehmer die Ausführungsrechte für eine Aktivität beanspruchen. Nachdem das SCI-Netzwerk die Aktion bestätigt hat, beginnt der Teilnehmer mit der Abarbeitung der Aufgabe.

Die Demonstration von dpex erfolgt über die Fortführung des in Kapitel 3 eingeführten Beispiels der Kollaboration einer Pizzabestellung. Dafür dient die in Abbildung 23 als BPMN-Prozessdiagramm dargestellte Beschreibung der Kollaboration als Grundlage. Das Modell wird im Folgenden noch einmal aufgearbeitet, erweitert und schließlich mit dpex ausgeführt.

6.5.3.1 *Modellierungsphase*

In der Pizzakollaboration werden mit Kunde, Pizza Place und Lieferdienst drei verschiedene Teilnehmer beziehungsweise Organisationen für das Prozessmodell identifiziert. In diesem Beispiel können verschiedene Kunden eine Pizza bei der Pizzeria Pizza Place bestellen, welche von einem Lieferanten von einem der beiden Lieferdienste Hermes oder Mercurius geliefert wird. Beim Pizza Place sind mehrere Pizzabäckerinnen beschäftigt sowie ein CEO. Nach Kapitel 2.5 müssen alle

möglichen Akteure der Prozessausführung in einem Organisationsmodell strukturiert werden. Die organisationalen Entitäten des Organisationsmodells werden anschließend im Prozessmodell referenziert, um die Bedingungen hinsichtlich der organisationalen Perspektive zu spezifizieren.

Prozessmodellierungssprache und Organisationsmodell

WAHL DER EXTERNEN KOMPONENTEN Zu Beginn müssen sich die Kollaborationsteilnehmer sowohl auf die Sprache für die Prozessmodellierung, die Struktur des globalen Organisationsmodells und auf das externe SCI-System einigen. In Kapitel 3.3 ist gezeigt, wie eine zwischenbetriebliche Kollaboration als BPMN-Prozessdiagramm dargestellt werden kann. Dementsprechend wird auch für die Kollaboration in dieser Demonstration ein BPMN-Prozessdiagramm als Artefakt eingesetzt. Die Möglichkeiten der Organisationsmodellierung werden in dieser Arbeit nicht im Detail betrachtet. Stattdessen orientiert sich die Kollaboration an dem tabellarischen Organisationsimplementierungsmodell der OrgEngine, wodurch den Prozessakteuren bestimmte Attribute zugeordnet werden können.

Wahl und Konfiguration der SCI

Als SCI einigen sich die Teilnehmer in dieser Demonstration auf das Ethereum-Protokoll. Da eine Ausführung auf der öffentlichen Ethereum-Blockchain mit höheren Kosten verbunden ist und das manuelle Betreiben eines Netzwerks zu aufwendig ist, verwenden die Teilnehmer mit *kaleido.io* einen Blockchain-as-a-Service-Dienst, um darüber eine eigene private Blockchain aufzusetzen. Inwieweit die Sicherheitsanforderungen durch den zentralisierten Anbieter einer dezentralen Blockchain eingeschränkt sind, soll hier nicht weiter diskutiert werden, da das Vorgehen lediglich die Funktionsweise des dpex-Frameworks demonstrieren soll. In der kostenlosen Demo-Version von *kaleido* können zwei Knoten erstellt werden, ein Knoten für Pizza Place und ein Knoten für die Lieferdienste. Für die Prozessausführung ist es entscheidend, dass die Prozessakteure Zugriffsrechte auf die Knoten haben, um ihre Transaktionen ins Netzwerk schicken zu können.

Unabhängigkeit von der Netzwerkkonfiguration

Tatsächlich ist es für dpex irrelevant, wie das Netzwerk aufgebaut ist und wie die Knoten über die teilnehmenden Organisationen verteilt sind. In der Theorie könnte das Netzwerk auch aus nur einem einzelnen Knoten bestehen, welcher beispielsweise vom Pizza Place verwaltet wird. Aus Sicherheitsgründen sollten die Daten und Rechte jedoch einen höheren Grad an Dezentralität aufweisen. Dieses DSR-Projekt fokussiert sich auf die Implementierung von dpex, wohingegen Konfigurationen externer Systeme lediglich im Einflussbereich des Frameworks betrachtet werden, sonst aber nicht weiter evaluiert werden sollen.

sciId und bpmId

ORGANISATIONSMODELL Die Ethereum-Blockchain dient der Kollaboration als SCI, wodurch das Sicherheitsmodul über die digitalen

Signaturen der Transaktionen im Ethereum-Netzwerk umgesetzt werden kann. Ähnlich zur Blockchain-basierten Prozessausführung wird der Prozessakteur, also der Initiator einer Prozessaktion, als Sender der entsprechenden Transaktion über den öffentlichen Schlüssel identifiziert, der zur Signierung der Transaktion verwendet wurde. Um zu vermeiden, dass die öffentlichen Schlüssel im Prozessmodell für die Modellierung der organisationalen Perspektive verwendet werden müssen, ordnet das Organisationsmodell den öffentlichen Schlüsseln als sogenannte *sciId*s eine sogenannte *bpmId* zu, welche dann im Prozessmodell verwendet wird.

Department und Role

Weiterhin muss im Organisationsmodell spezifiziert werden, zu welcher Organisation ein bestimmter Prozessakteur über die sciId und bpmId zuzuordnen ist (Tabelle 6). Da dies allein noch keine fortgeschrittenen Spezifikationen zulässt, können den Teilnehmern zusätzlich noch weitere beliebige Attribute zugeordnet werden, welche im Prozessmodell verwendet werden können.

bpmId	**sciId**	**Roles**	**Departments**
Christian	0xE076df...	Kunde	
Martin	0x4B1184...	Kunde	
Kerstin	0x33d329...	Pizzabäckerin	Pizza Place
Myriel	0x862C25...	Pizzabäckerin	Pizza Place
Stefan	0x422039...	CEO	Pizza Place
Lars	0x21aF08...	Lieferant	Hermes
ox7aFCdo	0x7aFCd0...	Lieferant	Mercurius

Tabelle 6: Organisationsmodell der Pizzakollaboration

Beispiel: Organisationsmodell. Im Beispielorganisationsmodell der Pizzakollaboration in Tabelle 6 sind zwei Kunden modelliert (Christian und Martin), welche auf der Blockchain jeweils einen Account mit dem öffentlichen Schlüssel 0xE076df... beziehungsweise 0x4B1184... besitzen. Kerstin und Myriel arbeiten als Pizzabäckerinnen im Restaurant im Pizza Place, welches von Stefan als CEO geführt wird. Für die Auslieferung der Waren sind zwei Lieferanten zuständig, wobei Lars für Hermes die Pizzen ausliefert. Der Lieferdienst Mercurius möchte die personenbezogenen Daten der Mitarbeiter auf zwischenbetrieblicher Ebene schützen, weswegen eine pseudonymisierte bpmId vergeben wird (letzte Zeile). Bei Bestehen eines berechtigten Interesses, etwa in einem Konfliktfall, wenn eine Lieferung nicht beim Kunden ankommt, ist es dennoch wichtig, dass der verantwortliche Mitarbeiter des Lieferdienstes ausfindig gemacht werden kann. Für die Konfliktlösung muss also entweder die Organisation selbst für

einen möglichen Schaden haften oder Mercarius muss die Identität des Mitarbeiters 0x7aFCd0... offenlegen.

Es ist zu beachten, dass das tabellenstrukturierte Organisationsmodell in Tabelle 6 eine primitive Form der Organisationsmodellierung darstellt. In der Praxis kommen komplexere Organisationsmodellierungssprachen zum Einsatz, mithilfe derer die organisationalen Strukturen realitätsgetreuer abgebildet werden [18]. Für den Einsatz in dpex mit den aktuell implementierten Konnektoren müssen die Organisationsmodelle in diese Tabellenform überführt werden. Der Aspekt der Organisationsmodellierung wird in dieser Arbeit nicht weiter betrachtet, stattdessen arbeitet dpex auf dem tabellenstrukturierten Implementierungsmodell.

Erweiterung um organisationale Konstrukte

PROZESSMODELL Das Prozessmodell aus Abbildung 23 wird für den praktischen Einsatz um die Komponente einer fortgeschrittenen Umsetzung der organisationalen Perspektive erweitert. Wie in Kapitel 2.5 dargestellt, eignet sich der BPMN-Standard dafür nur bedingt [135, S. 20]. In der Praxis kann im innerbetrieblichen Umfeld die organisationale Perspektive über die Funktionalitäten der WFMSs umgesetzt werden.[14] Um von herstellerspezifischen Konstrukten zu abstrahieren, unterstützt dpex die Spezifikation bestimmter organisationaler Konstrukte über BPMN-Annotationen.

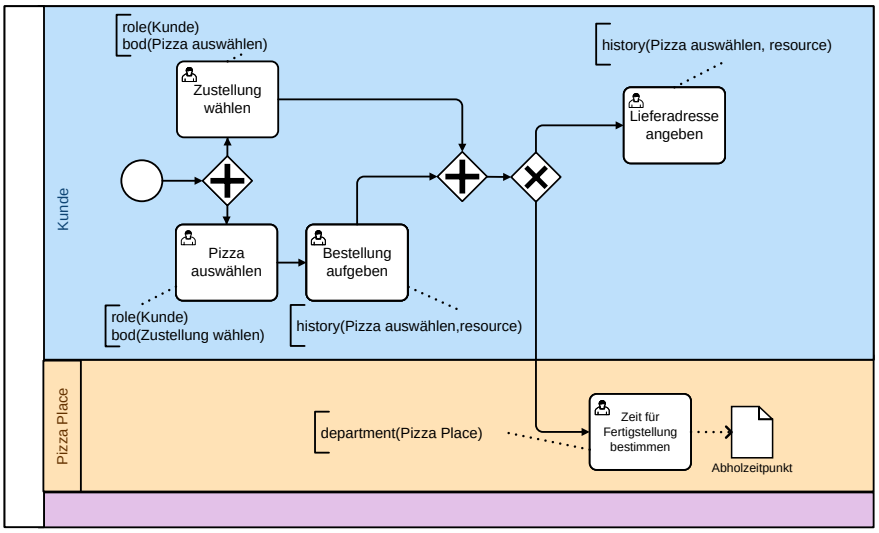

Abbildung 42: Umsetzung der organisationalen Perspektive in dpex am Beispiel eines Ausschnitts der Pizzakollaboration

14 https://docs.camunda.io/docs/components/best-practices/architecture/extending-human-task-management-c7/, besucht am 16.12.2023

Beispiel: Prozessmodell. Kapitel 3 diskutiert die Einzelheiten der Modellierung zwischenbetrieblicher Prozesse als BPMN-Prozessdiagramm am Beispiel der Pizzakollaboration. Für die Ausführung mit dpex sind die Aktivitäten mit Annotationen bezüglich der organisationalen Perspektive angereichert. Abbildung 42 zeigt einen Ausschnitt des Prozessmodells. Darin indiziert `role(Kunde)` in den Annotationen der Aktivitäten Zustellung wählen und Pizza auswählen, dass diese nur von Prozessakteuren bearbeitet werden dürfen, denen die entsprechende Rolle im Organisationsmodell zugeordnet ist. Weiterhin sorgt `bod(Pizza auswählen)` (*binding of duties*) bei Zustellung wählen dafür, dass die Aktivität Zustellung wählen vom gleichen Prozessakteur ausgeführt werden muss, der auch die Aktivität Pizza auswählen ausgeführt hat. Ähnlich sorgt die `history`-Bedingung an Lieferadresse angeben, dass diese Aktivität von der gleichen Ressource ausgeführt werden muss, welche Pizza auswählen ausgeführt hat. Wie in Kapitel 2.5.3 beschrieben, können `bod`, `history` oder beliebige weitere Makros in der OrgEngine implementiert werden.

6.5.3.2 *Implementierung der Konnektoren*

Bereitstellung im impl-Paket

Im Modellierungsschritt wurden sowohl das Prozess- und Organisationsmodell als auch Ethereum als externes System zur sicheren Kommunikation und Synchronisation von Nachrichten festgelegt. Im nächsten Schritt müssen nun die Konnektoren implementiert werden, um die externen Systeme zur Prozessausführung (Camunda) und für die sichere Nachrichtensynchronisierung (Ethereum) an dpex anzubinden. Dadurch dass die `dpex-lib` die Camunda- und Ethereum-Konnektoren im `impl`-Paket bereits zur Verfügung stellt (Abbildung 40), kann die Pizzakollaboration diese Klassen verwenden und muss die Konnektoren nicht manuell implementieren. Da dieser Schritt jedoch essenziell für die Anbindung weiterer externer Systeme ist, stellt das Kapitel dennoch diesen Implementierungsschritt für Konnektoren vor.

Flexible Wahl der Komponenten

CAMUNDA ADAPTER Da die Kollaboration mittels eines BPMN-Prozessdiagramms modelliert ist, müssen die Teilnehmer ein externes System in der BPM-Komponente zur Interpretation des BPMN-Modells anbinden. Dies kann für jede dpex-Instanz in den Organisationen lokal unabhängig voneinander entschieden werden. Um nur einen Adapter implementieren zu müssen, stimmen sich die Teilnehmer der Pizzakollaboration ab und binden mit dem Camunda WFMS jeweils das gleiche System an. Da Camunda die BPMN-Annotationen für die organisationale Perspektive nicht verarbeiten kann, wird das System über die Adapterimplementierung für die Anbindung an dpex in dieser Hinsicht über die in Kapitel 2.5.3 vorgestellte OrgEngine erweitert.

Adapterfabrik und Adapter

Für jeden Konnektor wird jeweils eine Adapterklasse (zum Beispiel `CamundaAdapter`) sowie eine zugehörige Fabrikklasse (zum Beispiel `CamundaAdapterFactory`) implementiert, welche jeweils die gleichen Felder für deren Konfiguration deklarieren (`String url;` in Abbildung 43, links). In der Mitte der Abbildung sind exemplarisch drei Instanzen der Fabrikklasse visualisiert, welche über ihre Objektfelder für verschiedene Camunda-Instanzen konfiguriert sind, etwa für ein lokales Camunda-System auf `localhost:8080` oder für ein fiktives extern bereitgestelltes Camunda-System auf `hosted-camunda.de`. Über die `create`-Methode der Fabriken kann eine `CamundaAdapter`-Instanz erzeugt werden, welche dann ebenfalls mit der URL der entsprechenden Fabrikinstanz konfiguriert ist. Dies erfolgt später automatisiert bei der Konfiguration der Kollaboration in Kapitel 6.5.3.4.

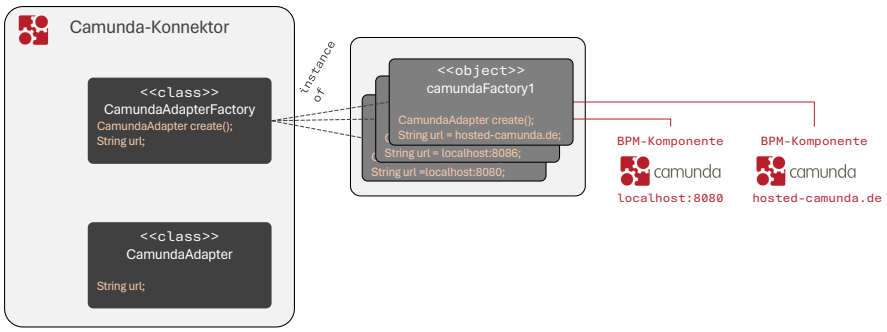

Abbildung 43: Instanziierung von `CamundaAdapterFactory`

Implementierung der Fabrik

Quelltext 11 und Quelltext 12 zeigen auszugsweise die Implementierung der abgebildeten Klassen. Die `CamundaAdapterFactory` erweitert in Quelltext 11 die Klasse `BPMEngineFactory`, welche eine abstrakte Methode `create` zur Erzeugung einer Adapterinstanz definiert. In der implementierenden `create`-Methode wird zuerst eine Instanz des Adapters erstellt (Zeile 6), bevor die nötigen Objektfelder gesetzt werden (Zeilen 7 und 8). Anschließend wird das gegebene Prozessmodell im externen Camunda WFMS bereitgestellt (Zeile 9) und die Adapterinstanz schließlich zurückgegeben (Zeile 10).

```
1  public class CamundaAdapterFactory extends BPMEngineFactory {
2
3      private String camundaUrl;
4
5      public CamundaAdapter create(ProcessModel processModel) {
6          CamundaAdapter ca = new CamundaAdapter();
7          ca.setCamundaUrl(this.camundaUrl);
8          ca.setName(super.getName());
9          ca.deploy(((CamundaModel) processModel).getBpmnModel());
10         return ca;
11     }
12 }
```

Quelltext 11: Implementierung von `CamundaAdapterFactory`

Implementierung des Adapters

Quelltext 12 zeigt auszugsweise die Implementierung des Adapters. In Zeile 10 wird mit der `claim`-Methode die Umsetzung einer Benutzeraktion zur Anforderung der Ausführungsrechte demonstriert. Nachdem die benötigten Informationen aus den Übergabeparametern extrahiert sind (Zeilen 11 bis 13), wird in den Zeilen 14 und 15 die URL konstruiert und die HTTP-Anfrage an das externe Camunda-System zur Aktualisierung des Prozessstatus geschickt.

OrgEngine-Integration

Die Erweiterung von Camunda hinsichtlich der organisationalen Perspektive erfolgt über die `isConform`-Methode (Zeile 18), worin die OrgEngine aufgerufen wird. Der Aufruf von `isConform` wird vom ICB initiiert (Kapitel 6.5.3.6).

```
1  public class CamundaAdapter extends BPMEngine {
2
3      HttpClient httpClient = HttpClient.newHttpClient();
4
5      public CamundaAdapter(String camundaUrl) {
6          this.camundaUrl = camundaUrl; }
7
8      Instance instantiate(String modelRef, String name, String variables) { ... }
9
10     void claim(Instance instance, TaskAction taskAction) {
11         String taskId = this.getTaskIdFromTaskName(
12             taskAction.getTask().getActivity(),
13             instance.getLocalInstanceReference());
14         String executeUrl = this.camundaUrl + "task/" + taskId + "/claim";
15         this.sendRequest(executeUrl, "{\"userId\": \"" + taskAction.getUser().getBpmId()
               + "\"}");
16     }
17
18     public boolean isConform(Instance i, TaskAction taskAction) { orgEngine.evaluate(...) ...
           }
19
20     private DpexHttpResponse sendRequest(String url, String body) { ... }
21 }
```

Quelltext 12: Implementierung von `CamundaAdapter`

Anbindung von Ethereum an dpex

ETHEREUM SCI Bevor ein Adapter für die Ethereum-Blockchain im dpex-Framework beschrieben wird, muss zuerst ein Konzept für deren Anbindung festgelegt werden. In Kapitel 6.4 wird die SCI als externe Komponente beschrieben, welche ungeordnete, asynchrone Nachrichten von Prozessteilnehmern synchronisiert. Die Ethereum-Blockchain erlaubt jedem Teilnehmer das Senden von Transaktionen ins Netzwerk, welche über bestimmte Algorithmen mit der Blockchain-Datenstruktur in eine global akzeptierte Reihenfolge gebracht werden. Gerade auf einer öffentlichen Blockchain können pro Tag allerdings über eine Million Transaktionen verifiziert werden[15], sodass ein manuelles Filtern und Verarbeiten aller Transaktionen unpraktikabel ist. Stattdessen nutzt das dpex-Framework einen Smart Contract, welcher die für die Prozessausführung relevanten Transak-

15 `https://etherscan.io/txs`, besucht am 17.12.2023

tionen kapselt. Das folgende Konzept ist nur eine Möglichkeit der Umsetzung, während andere Kollaborationen die Anbindung einer Ethereum-Blockchain auf andere Weise konzeptionieren können.

Konzeptionelle Umsetzung

Die Umsetzung erfolgt über zwei verschiedene Smart Contracts, `DpexSci` (Quelltext 13) und `DpexProcessInstance` (Quelltext 14), welche über das Aussenden von Ereignissen die Netzwerkkomponenten der dpex-Anwendungen benachrichtigen können. Im Gegensatz zur Blockchain-basierten Prozessausführung (Kapitel 5) setzen diese Smart Contracts keine Prozesslogik um, sondern dienen lediglich dazu, relevante Transaktionen zu gruppieren und Ereignisse auszusenden. Die Aufgabe des `DpexSci`-Smart Contracts ist es dabei, alle Teilnehmer über neue Prozessinstanzen zu informieren. Alle Aktionen bezüglich einer Prozessinstanz werden dann über einen spezifischen `DpexProcessInstance`-Smart Contract kommuniziert.

Kapselung von Transaktionen statt Interpretation von Prozessen

DpexSci

Im vorgeschlagenen Konzept existiert für jede Kollaboration, also für jedes `Alliance`-Objekt, ein einziger, eigener `DpexSci`-Smart Contract, welcher global einmal über die dpex-Anwendung bereitgestellt werden muss. Die resultierende Adresse wird dann zu Konfigurationszwecken jedem Kollaborationsteilnehmer mitgeteilt.

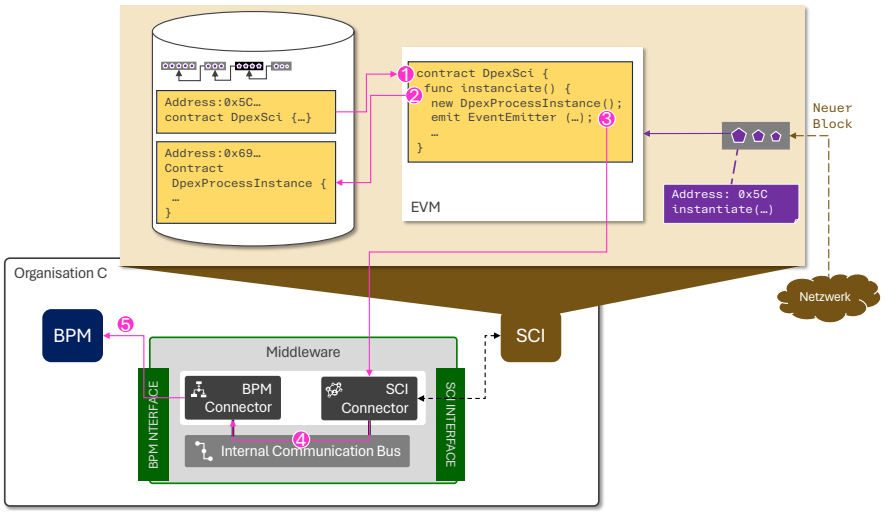

Abbildung 44: Schematische Darstellung der Integration der Ethereum-Blockchain als SCI an das dpex-Framework

Prozessinstanziierung

Abbildung 44 zeigt die technische Funktionsweise und den Ablauf des Konzepts zur Ethereum-Integration anhand der Instanziierung eines Prozessmodells. Das Abschließen von Aktivitäten folgt einem ähnlichen Schema. Der `DpexSci`-Smart Contract ist bereits bereitgestellt und somit in jeder lokalen Kopie der Blockchain (gelbes Rechteck) unter der Adresse `0x5C...` verfügbar. Im abgebildeten Szenario erreicht gerade ein neu verifizierter Block den lokalen Ethereum-Client, welcher daraufhin die darin enthaltenen Transaktionen *ausführt*, um die lokale Blockchain-Kopie zu aktualisieren. Im Block findet sich eine Transaktion, welche einen Aufruf der `instantiate`-Funk-

tion des `DpexSci`-Smart Contracts mit der Adresse `0x5C...` darstellt (violett in Abbildung 44 rechts). Die Abarbeitung dieser Transaktion ist in pinken Ziffern angedeutet. Zuerst wird der `DpexSci`-Smart Contract (`0x5C...`) in die EVM geladen, um die `instantiate`-Funktion auszuführen ①. Darin wird eine neue Instanz des `DpexProcessInstance`-Smart Contracts erzeugt ② (Quelltext 13, Zeile 9). Das bedeutet, dass in der Blockchain-Datenstruktur ein Speicherbereich für diesen Smart Contract reserviert wird, welcher dann über eine deterministisch berechnete, also auf allen Knoten identische Smart Contract-Adresse angesprochen werden kann (`0x69...`). Anschließend sendet der `DpexSci`-Smart Contract ein Ereignis aus (Quelltext 13, Zeile 11). Das Ereignis beinhaltet die Adresse des neu erzeugten `DpexProcessInstance`-Smart Contracts sowie einen benutzerdefinierten Namen der Instanz und gegebenenfalls initialen Prozessvariablen (Quelltext 13, Zeilen 3 und 4). Durch einen registrierten Event-Handler werden die Ethereum-SCI-Konnektoren der dpex-Anwendungen über diese Events benachrichtigt ③. Für den hier abgebildeten Fall einer Instanziierung wird anschließend über die Middleware der entsprechende BPM-Konnektor der Kollaboration geladen ④, welcher für die Instanziierung des entsprechenden Prozessmodells in der externen BPM-Komponente sorgt ⑤.

Laden von DpexSci in EVM

Instanziierung von DpexProcssInstance

Aussenden des Ereignisses

Aufruf der SCI-Konnektoren

Laden des BPM-Konnektors

Prozessinstanziierung

```
1  contract DpexSci {
2
3      event EventEmitter(
4          DpexProcessInstance processInstanceContract, string variables, string name);
5      DpexProcessInstance[] public instances;
6
7      function instantiate(
8              string memory variables, string memory name) public returns (address) {
9          DpexProcessInstance dpexProcessInstance = new DpexProcessInstance();
10         instances.push(dpexProcessInstance);
11         emit EventEmitter(dpexProcessInstance, variables, name);
12         return address(dpexProcessInstance);
13     }
14 }
```

Quelltext 13: Smart Contract, der auf Ethereum zur Verwaltung von Kollaborationen in dpex bereitgestellt werden muss

Prozessausführung

Der bereitgestellte `DpexProcessInstance`-Smart Contract steht nun stellvertretend für die neue Prozessinstanz. Alle Aktionen der Prozessakteure werden als Transaktion, welche die `store`-Funktion des entsprechenden Smart Contracts aufrufen, repräsentiert. Die `store`-Funktion wird dabei mit dem Namen der Aktivität, der Stufe des Aktivitätslebenszyklus (`claim` oder `complete`) sowie gegebenenfalls den Prozessvariablen als Parameter aufgerufen (Quelltext 14, Zeilen 11 und 12). Ähnlich zur Instanziierung von Prozessmodellen sendet die `store`-Funktion ein Ereignis (Zeile 15), wofür im Netzwerkmodul des Ethereum-Adapters ebenfalls ein Event-Handler registriert

```
1  contract DpexProcessInstance {
2      struct Event {
3          string activity;
4          string stage;
5          address resource;
6          string variables;
7      }
8      event EventEmitter(Event logEntry);
9      Event[] log;
10
11     function store(
12             string memory activity, string memory stage, string memory variables) public {
13         Event memory logEntry = Event(activity, stage, msg.sender, variables);
14         log.push(logEntry);
15         emit EventEmitter(logEntry);
16     }
17 }
```

Quelltext 14: Smart Contract, der auf Ethereum zur Verwaltung von Prozessinstanzen in dpex bereitgestellt wird

ist. Ähnlich zum in Abbildung 44 dargestellten Ablauf kann das dpex-Framework darüber die lokalen WFMSs benachrichtigen.

ETHEREUM ADAPTER Analog zur BPM-Domäne müssen für den Adapter zur Anbindung der Ethereum-Blockchain sowohl eine `EthereumAdapterFactory` als auch ein `EthereumAdapter` implementiert werden. Der Adapter selbst wird mit jeweils einem Netzwerkmodul (`EthereumNetwork`), einem Sicherheitsmodul (`NoSecurity`) und einem Synchronisierungsmodul (`TrueConsensus`) konfiguriert, sodass auch diese Module implementiert werden müssen.

Sicherheit und Synchronisierung

Die Ethereum-Blockchain erfüllt bereits die Anforderungen hinsichtlich Sicherheit (Signierung der Transaktionen) und Synchronisierung (Ordnen der Transaktionen in Blöcken)[16], sodass einfache Attrappenimplementierungen der Module verwendet werden können. In Quelltext 15 sind auszugsweise die `NoSecurity`-Klasse und die `TrueConsensus`-Klasse abgebildet. Die Methoden `verify` und `isAgreementReached` geben dabei stets `true` zurück, statt komplexere Programmlogik zu implementieren. Der Einsatz der Module wird bei der Beschreibung der Prozessausführung mit dpex in Kapitel 6.5.3.5 gezeigt.

Netzwerk

Das Netzwerkmodul ist ausschnittsweise in Quelltext 16 dargestellt und verwendet die *Web3j*-Bibliothek für die Kommunikation mit einem Ethereum-Knoten. Im Quelltext ist die Methode `sendInstantiation` exemplarisch dargestellt. Sie wird aufgerufen, wenn ein Benutzer die Instanziierung eines Prozessmodells initiiert (vergleiche Abbildung 44). Zuerst wird in den Zeilen 9 und 10 das Web3j-Objekt initialisiert. Hierfür wird die URL-Adresse benötigt, über welche der externe Blockchain-Knoten erreichbar ist. Die URL ist ähnlich zum

16 Zumindest bei deterministischer Finalität. Die Probabilistische Finalität wird später diskutiert.

```
1  public class NoSecurity extends Security {
2      public boolean verify(String text, String signature, String publicKey) {
3          return true; }
4  ...
5  public class TrueConsensus extends ConsensusEngine {
6      public boolean isAgreementReached(Ack a, CandidateLog clog, String... values) {
7          return true; }
8  ...
9  }
```

Quelltext 15: Implementierung von `NoSecurity` und `TrueConsensus`

`CamundaAdapter` über die Fabrik konfiguriert und kann über `getSciAddress` abgerufen werden. In Zeile 11 wird der `DpexSci`-Smart Contract über eine Java-Wrapperklasse geladen. Über die `globalAllianceReference` des `instantiation`-Objekts kann die Adresse des Smart Contracts identifiziert werden (Zeile 12)[17]. In den Zeilen 13 bis 15 wird der Aufruf der `instantiate`-Funktion mit den entsprechenden Parametern auf dem Smart Contract vorbereitet. Die Zeile 16 sorgt dafür, dass die entsprechende Transaktion an das Blockchain-Netzwerk geschickt wird. Damit endet der Programmfluss. Die Instanziierung wird wie in Abbildung 44 beschrieben fortgesetzt, sobald diese Transaktion asynchron in einem Block verifiziert ist.

```
1  public class EthereumNetwork extends Network {
2
3      public EthereumRawNetwork(String sciContractAddress) {
4          EthFilter filter = new EthFilter(EARLIEST, LATEST, sciContractAddress);
5          sciHandler = web3.ethLogFlowable(filter).subscribe(this::handleInstantiation);
6      }
7
8      public void sendInstantiation(Instantiation instantiation) {
9          HttpService httpService = new HttpService(super.getSciAddress());
10         Web3j web3 = Web3j.build(httpService);
11         DpexSci dpexSci = DpexSci.load(
12             instantiation.getGlobalAllianceReference(), web3, ...);
13         rfc = dpexSci.getClass()
14             .getMethod("instantiate", String.class, String.class)
15             .invoke(dpexSci, instantiation.getJSONVariables(), instantiation.getName());
16         rfc.send();
17     ...
```

Quelltext 16: Implementierung des Netzwerkmoduls für den Ethereum-Adapter

Initialisierung des Netzwerks

Im Konstruktor des Netzwerkmoduls des Ethereum-Adapters wird dem oben vorgestellten Konzept entsprechend ein Event-Handler auf die Ereignisse des `DpexSci`-Smart Contracts registriert. Dies wird im Quelltext 16 in den Zeilen 3 bis 6 beschrieben. Dort wird die `handleInstantiation`-Methode der `EthereumNetwork`-Klasse als Event-Handler für Events des Smart Contracts registriert, welcher über die Adresse `sciContractAddress` erreichbar ist (Zeile 5). Die `sciContractAd-`

17 Als Konvention wird die `DpexSci`-Smart Contract-Adresse als Wert für die `globalAllianceReference` angegeben.

dress ist bei der Erstellung des Alliance-Objekts zu übergeben (Kapitel 6.5.3.4). Die Implementierung der weiteren Funktionen des Netzwerkmoduls wird bei der Präsentation des Programmablaufs in Kapitel 6.5.3.5 detaillierter beschrieben.

PROZESSMODELL Zuletzt muss auch für das Prozessmodell eine Fabrik und ein Adapter implementiert werden, sodass diese zur Konfiguration eines Alliance-Objekts verwendet werden können. Quelltext 17 zeigt die Implementierung des Adapters CamundaMPModel. Der Adapter und die dazugehörige Fabrikklasse wird mit einem eindeutigen Namen modelReference konfiguriert (Zeile 4) sowie dem Prozessmodell als serialisiertes Bytearray (Zeile 6) und dem Organisationsmodell als Zeichenkette (Zeile 7).

```
1 public class CamundaMPModel extends ProcessModel {
2
3     // Vererbt von ProcessModel
4     String modelReference;
5
6     byte[] bpmnModel;
7     String organizationalModel;
8  ...
9 }
```

Quelltext 17: Implementierung des multiperspektivischen Prozessmodell-Adapters

6.5.3.3 *Vorbereitungen und Installation*

Nachdem das Prozess- und Organisationsmodell sowie der *kaleido* Blockchain-as-a-Service-Dienst als Artefakte festgelegt und die entsprechenden Adapter für deren Anbindung an dpex implementiert sind, kann die benötigte Software installiert und die notwendigen externen Dienste konfiguriert werden. Wie bereits erwähnt, sind die Konnektoren für Ethereum und Camunda in dpex-lib schon im impl-Paket implementiert, sodass die Schritte des vorherigen Kapitels für die Pizzakollaboration und für alle weiteren Kollaborationen, welche Camunda und Ethereum verwenden, übersprungen werden können.

drei Dienste

Im Rahmen des dpex-Frameworks verwaltet jeder Teilnehmer grundsätzlich drei lokale Dienste. Zum einen wird ein WFMS zur Prozessinterpretation benötigt. Andererseits muss ein Dienst für die SCI bereitgestellt werden, etwa ein Knoten im Ethereum-Netzwerk. Bei der Verwendung von *kaleido* ist dieser Dienst ausgelagert und muss nicht lokal verwaltet werden. Zuletzt betreibt jeder Prozessteilnehmer eine dpex-Instanz.

Start über Docker

Das dpex-build-Projekt stellt eine docker-compose-Datei bereit, über welche die dpex-Anwendung gestartet werden kann. Als Voraussetzung ist damit lediglich der Containerorchestrierungsdienst Docker

zu installieren. Das Projekt ist über *GitLab*[18] verfügbar und kann direkt über einen Link[19] heruntergeladen werden. Die Docker-Container können wie üblich über den folgenden Befehl initialisiert und gestartet werden:

```
docker-compose up
```

Nachdem der Befehl ausgeführt ist, kann die dpex-Instanz über einen Browser unter der URL `localhost:3000` aufgerufen werden.

Bereitstellung aller Komponenten in Docker

Für die Demonstration müssen keine zusätzlichen Komponenten installiert werden. Zum einen wird in der `docker-compose`-Datei konfiguriert, dass eine Camunda-Instanz als extra Docker-Container mit hochgefahren wird.[20] Zum anderen verwendet die Pizzakollaboration den Blockchain-as-a-Service-Dienst *kaleido* als SCI, sodass auch hier keine lokalen Ethereum-Knoten bereitgestellt werden müssen. Als Alternative wird über die `docker-compose`-Datei ein weiterer Container gestartet, welcher die Blockchain-Simulation *ganache* zur Verfügung stellt. Dadurch ist über `ganache:8545` innerhalb des Docker-Netzwerks ein Blockchain-Knoten eines simulierten Ethereum-Netzwerkes zu lokalen Demonstrationszwecken verfügbar.

6.5.3.4 *Konfiguration der `Alliance`*

Innerhalb der lokal laufenden dpex-Anwendung muss nun jeder Teilnehmer ein sogenanntes `Alliance`-Objekt konfigurieren. Dabei sind unter anderem die zu verwendenden Adapter und Adapterfabriken auszuwählen.

INSTANZIIERUNG DER ADAPTER-FABRIKEN Bevor die zwischenbetriebliche Kollaboration als `Alliance` konfiguriert werden kann, müssen zuerst die zu verwendenden Adapterfabriken instanziiert werden.

Auswahl eines Konnektors

Dafür wird dem Benutzer im Frontend die Auswahl an zur Verfügung stehenden Konnektoren angezeigt (pinkes Rechteck in Abbildung 45).

Eingabe der Konfigurationsparameter

Sobald ein Konnektor zur Instanziierung ausgewählt ist, werden vom Benutzer die benötigten Konfigurationsparameter abgefragt, mit denen eine Instanz der Adapterfabrik erzeugt werden kann. Im Szenario in Abbildung 45 sind drei verschiedene Fabrikinstanzen angelegt, welche jeweils einer bestimmten Camunda-Instanz, identifiziert durch die URL, zuzuordnen sind.

Wenn analog eine `EthereumAdapterFactory` instanziiert ist, können nun die entsprechenden Adapterinstanzen erzeugt werden. Dies erfolgt automatisiert über die Erzeugung und Konfiguration eines `Alliance`-Objekts, worüber alle für die Abwicklung der zwischenbetrieblichen Kollaboration notwendigen Informationen gekapselt sind.

18 https://gitlab.com/bpm-dpex/dpex-build/-/tree/v0.2, besucht am 08.03.2024
19 https://gitlab.com/bpm-dpex/dpex-build/-/archive/v0.2/dpex-build-v0.2.zip, besucht am 08.03.2024
20 https://hub.docker.com/r/camunda/camunda-bpm-platform/, besucht am 06.01.2023

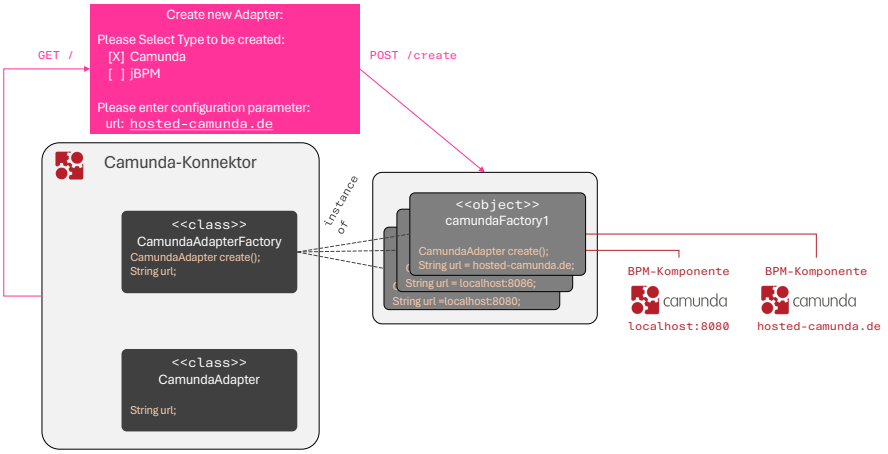

Abbildung 45: Auswahl und Instanziierung einer `CamundaAdapterFactory`

INSTANZIIERUNG DER ADAPTER UND KONFIGURATION EINER ALLIANCE Zur Erstellung einer `Alliance` werden dem Benutzer alle zur Verfügung stehenden Fabrikinstanzen zu allen Konnektoren angezeigt (pinkes Rechteck in Abbildung 46). Der Benutzer entscheidet sich durch die Auswahl einer bestimmten Fabrikinstanz für die Verwendung der referenzierten externen Komponente. Im Beispiel in Abbildung 46 wählt der Benutzer die Fabrikinstanz aus, welche mit dem Wert `hosted-camunda.de` konfiguriert ist. Bei der Erstellung der `Alliance` durch dpex wird jetzt die `create`-Methode dieser Fabrikinstanz aufgerufen, welche eine neue `CamundaAdapter`-Instanz unter Verwendung der jeweiligen Konfigurationsparameter erzeugt. Durch die Zuweisung dieser Adapterinstanz im `Alliance`-Objekt, verwendet diese `Alliance` nun das Camunda-System, welches unter `hosted-camunda.de` erreichbar ist.

Auswahl einer Fabrikinstanz

Automatisierte Instanziierung eines Adapters

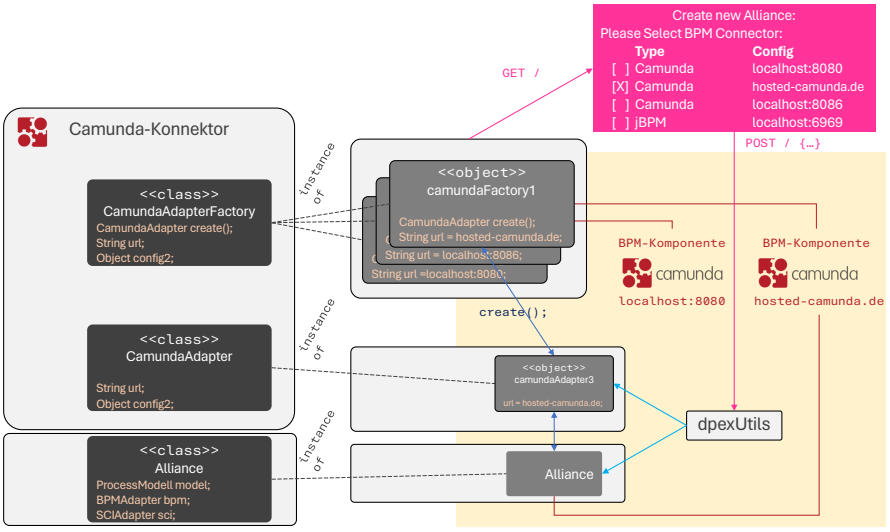

Abbildung 46: Auswahl und Instanziierung eines `CamundaAdapters`

VOLLSTÄNDIGE KONFIGURATION DER ALLIANCE Für die vollständige Konfiguration einer `Alliance` müssen die Teilnehmer die benötigten Adapter für das Prozessmodell, den BPM-Konnektor und den SCI-Konnektor selektieren. Dabei unterstützt die Hilfsklasse `DpexUtils`, welche in `dpex-lib` implementiert ist. Die Klasse stellt eine Methode zur Verfügung, welche Fabrikinstanzen anhand ihres Namens aus der Datenbank lädt, automatisiert die gewählten Adapter erstellt und ein `Alliance`-Objekt damit konfiguriert.

buildAlliance in DpexUtils

```
1  public Alliance buildAlliance(String name, String _model, String _engine, String _sci, String
       gar, SCIConfig sciConfig) {
2
3      Alliance alliance = new Alliance();
4      alliance.setName(name);
5
6      ProcessModel model = processModelFactoryRepo.findById(_model).create();
7      BPMEngine engine = BPMEngineFactoryRepo.findById(_engine).create(model);
8      Collaboration sci = collaborationFactoryRepo.findById(_sci).create(sciConfig);
9
10     alliance.setModel(model);
11     alliance.setEngine(engine);
12     alliance.setCollaboration(sci);
13     alliance.setGlobalAllianceReference(gar);
14
15     return alliance;
16 }
```

Quelltext 18: Erstellung einer Alliance

Diese Methode `buildAlliance` ist in Quelltext 18 abgebildet und setzt voraus, dass die gewählten Fabrikinstanzen bereits erzeugt sind. Über die Methodenparameter (Zeile 1) werden die zu konfigurierenden Adapter ausgewählt. Nachdem in den Zeilen 3 und 4 ein `Alliance`-Objekt erzeugt und der gegebene Name gesetzt ist, werden in den Zeilen 6 bis 8 die Adapterfabriken über die jeweils als Parameter übergebenen Namen aus der Datenbank geholt und die `create`-Methode auf den Fabrikinstanzen aufgerufen. Dabei wird die Prozessmodelladapterinstanz bei der Erstellung der Adapterinstanz für das WFMS als Parameter übergeben, damit das Prozessmodell dort bereitgestellt werden kann. Für die Instanziierung eines SCI-Adapters wird eine `SCIConfig` übergeben. Bei der Verwendung von Ethereum als SCI enthält die `EthereumConfig` die Adresse des `DpexSci`-Smart Contracts, über dessen Ereignisse die neuen Instanzen empfangen werden können. Bei der Instanziierung einer `EthereumNetwork`-Instanz in der `create`-Methode der Ethereum-Adapterfabrik (Quelltext 18, Zeile 8) wird auf Basis dieser Adresse ein Event-Handler registriert. Diese Vorgehensweise erlaubt die Wiederverwendung des Ethereum-Adapters für verschiedene Kollaborationen, welche verschiedene `DpexSci`-Smart Contracts verwenden.

BPM-Konnektor

SCI-Konnektor

GAR

Im Anschluss werden dem `Alliance`-Objekt noch die Adapterinstanzen sowie eine `GlobalAllianceReference (GAR)` zugewiesen, bevor die `Alliance` zurückgegeben wird. Die `GAR` ist ein kollaborations-

weit eindeutiger Schlüssel, über welchen die `Alliance`-Objekte identifiziert und zur Kollaboration zugeordnet werden können. Während die `GAR` prinzipiell beliebig gewählt werden kann, gilt in der aktuellen Implementierung des Ethereum-Adapters die Konvention, die Smart Contract-Adresse des `DpexSci`-Smart Contracts als `GAR` zu verwenden. Diese Adresse muss also beim Erzeugen der `Alliance`-Objekte bekannt sein. Das `dpex-react`-Frontend unterstützt die Funktionalität, den `DpexSci`-Smart Contract auf einer Ethereum-Blockchain bereitzustellen. Die dadurch generierte Adresse muss anschließend den Kollaborationsteilnehmer für die Konfiguration der `Alliance` übermittelt werden.

6.5.3.5 *Instanziierung eines Prozessmodells*

Nach dem Implementieren der Konnektoren, dem Anlegen der Adapterfabriken und dem Konfigurieren eines `Alliance`-Objekts, worüber die Adapterfabriken instanziiert werden, startet die Prozessausführung mit dpex ähnlich der zentralisierten Prozessausführung mit der Instanziierung des Prozessmodells. Dabei schränkt dpex in der aktuellen Implementierung nicht ein, welcher Prozessteilnehmer die Instanziierung veranlassen darf.

Wie alle Aktionen müssen auch Instanziierungen asynchron von der SCI-Komponente bestätigt werden. Damit teilt sich der Kontrollfluss aller Aktionen in das Senden von Nachrichten beziehungsweise Transaktionen und in das Empfangen oder Behandeln von externen Ereignissen, sobald die Transaktionen bestätigt sind.

sendInstantiation-Methode

SENDEN EINER INSTANZIIERUNGSTRANSAKTION Sobald ein Benutzer über die Benutzerschnittstelle die Instanziierung veranlasst, wird im ICB, also im `DPEXService`, das `Instantiation`-Objekt in dem Sicherheitsmodul signiert, welches über die `Alliance` für die entsprechende Kollaboration konfiguriert ist (Quelltext 19, Zeilen 4 und 5). In den Zeilen 6 und 7 wird das Netzwerkmodul geladen und die `sendInstantiation`-Methode darauf aufgerufen (Zeile 8).

```
1  public class DPEXService {
2      ...
3      public void sendInstantiation(Alliance alliance, Instantiation instantiation) {
4          Security security = alliance.getSecurity();
5          security.sign(instantiation);
6          Collaboration collaboration = alliance.getCollaboration();
7          Network network = collaboration.getNetwork();
8          network.sendInstantiation(instantiation);
9      }
10     ...
11 }
```

Quelltext 19: Verarbeitung der Instanziierungsaktion im ICB `DPEXService`

Das tatsächliche Senden der Transaktion über eine Instanz des `EthereumNetwork`-Moduls ist oben in Quelltext 16 bereits beschrieben.

handleInstantiation-Methode

BEHANDELN EINER INSTANZIIERUNGSTRANSAKTION Das Behandeln einer Instanziierungsaktion ist in Abbildung 44 in den Schritten ③, ④ und ⑤ angedeutet. Quelltext 20 zeigt die Umsetzung im Ethereum-Adapter, der im Folgenden näher erläutert wird.

```
1  public class EthereumNetwork extends Network {
2      ...
3      public void handleInstantiation(Object event) {
4          // Parse data from the event and build the instantiation object
5          Log log = (Log) event;
6          List<Type> args = FunctionReturnDecoder.decode(log.getData(), DpexSci.
               EVENTEMITTER_EVENT.getParameters());
7          String globalAllianceReference = log.getAddress();
8          String globalInstanceReference = args.get(0).toString();
9          String JSONVariables = args.get(1).toString();
10         String name = args.get(2).toString();
11         Instantiation inst = new Instantiation(globalAllianceReference,
               globalInstanceReference, "", "", JSONVariables, name);
12
13         // Register the handleTaskAction-Method as event listener for
               DpexProcessInstance events
14         HttpService httpService = new HttpService(super.getSciAddress());
15         Web3j web3 = Web3j.build(httpService);
16         EthFilter filter = new EthFilter(EARLIEST, LATEST, inst.
               getGlobalInstanceReference());
17         bpmHandler = web3.ethLogFlowable(filter).subscribe(this::handleTaskAction);
18
19         ApplicationContextProvider.bean(DPEXService.class).handleInstantiation(inst);
20     }
21 }
```

Quelltext 20: Behandeln einer eingehenden Instanziierungsaktion im `EthereumNetwork`

Kapitel 6.5.3.2 beschreibt bei der Implementierung des Ethereum-Adapters im Quelltext 16 in Zeile 5 wie die Methode `handleInstantiation` (Quelltext 20) als Event-Handler für Instanziierungstransaktionen registriert wird. Dementsprechend wird die Methode automatisch aufgerufen, wenn während der Ausführung der `instantiate`-Funktion in einem `DpexSci`-Smart Contract im Ethereum-Client ein solches Ereignis ausgesendet wird.

In den Zeilen 5 bis 10 werden aus dem `event`-Parameter die einzelnen Felder gelesen. Dabei wird in Zeile 7 die `GAR` extrahiert, also die Adresse des `DpexSci`-Smart Contracts, der das empfangene Ereignis ursprünglich aussendete. Über die `GAR` kann im weiteren Verlauf das zugehörige `Alliance`-Objekt in der lokalen Datenbank identifiziert werden. Die `globalInstanceReference (GIR)` wird in Zeile 8 aus dem ersten Parameter des Ereignisses gelesen, dem die Adresse des neu erstellten `DpexProcessInstance`-Smart Contracts zugewiesen ist. Nachdem weitere Informationen analysiert und das `Instantiation`-Objekt in Zeile 11 erstellt ist, wird anschließend ein Event-Handler

für die Events des neu erstellten `DpexProcessInstance`-Smart Contracts registriert. Dabei wird die `GIR` in Zeile 16 als Quelle angegeben und die `handleTaskAction`-Methode in Zeile 17 als Event-Handler-Methode registriert. Dies sorgt dafür, dass das Netzwerkmodul über neue Ereignisse im `DpexProcessInstance`-Smart Contract benachrichtigt und die entsprechende Methode automatisch aufgerufen wird. Letztendlich wird in Zeile 19 die Programmkontrolle an den `DPEXService` abgegeben, worin über das `Alliance`-Objekt der konfigurierte BPM-Konnektor geladen und darauf die `instantiate`-Methode aufgerufen wird, um eine neue Instanz im externen WFMS zu erzeugen.

6.5.3.6 *Aktualisieren des Lebenszyklusstatus einer Aktivität*

Der Ablauf zum Aktualisieren des Lebenszyklusstatus einer Aktivität folgt in ähnlicher Weise der oben beschriebenen asynchronen Kommunikation. Das heißt, dass jede Benutzeraktion, etwa die Beanspruchung der Ausführungsrechte für eine Aktivität, als Transaktion in das Netzwerk geschickt wird, womit der Programmablauf ohne Auswirkung auf den Prozessstatus endet. Nach der globalen Synchronisation wird ein neuer Programmfluss gestartet, in dem das Ereignis verarbeitet wird.

zweistufiges Verfahren

Wie bei der Darstellung der allgemeinen Kommunikationsschemata in Kapitel 6.4.5 beschrieben, verfolgt das dpex-Framework ein zweistufiges Verfahren für das Empfangen und Verarbeiten von Aktionen Dies erlaubt eine manuelle SCI-Implementierung, bei der beim Empfang einer Nachricht noch nicht deren endgültige Bestätigung und globale Ordnung eindeutig bestimmt werden kann und gegebenenfalls auf weitere Nachrichten, etwa Bestätigungsnachrichten, gewartet werden muss. Dieses Prinzip wird auch in der Implementierung des Frameworks in `dpex-lib` berücksichtigt, wobei bei der Anbindung von Ethereum als SCI bestimmte Vereinfachungen erzielt werden können, da die ankommenden Nachrichten bereits final bestätigt sind. Die allgemeine Implementierung sowie der Spezialfall Ethereum sollen nun anhand der Prozessaktion für das Beanspruchen der Ausführungsrechte skizziert werden.

Senden einer Prozessaktion

ALLGEMEINER ABLAUF Die Aufrufsequenz in dpex für das Senden einer Prozessaktion ist in Abbildung 47 dargestellt. Nach dem Auswählen der Aktion in der Benutzerschnittstelle (1.), wird die Kontrolle an die `sendTaskAction`-Methode im `DPEXService` abgegeben (2.). Über die `GAR` und das darüber identifizierte `Alliance`-Objekt werden die BPM- und SCI-Komponenten geladen. Zuerst wird über die `isConform`-Methode der BPM-Komponente die Konformität der Aktion bezüglich des lokalen Prozessstatus überprüft, um nicht konforme Transaktionen zu vermeiden (3.), wobei die Überprüfung zusätzlich auch beim Empfang einer Nachricht erfolgt (siehe unten). Die `isConform`-Methode wird vom BPM-Adapter bereitgestellt und kann

neben einfachen Anfragen an das externe WFMS auch zusätzliche Logik, wie am Beispiel der OrgEngine beschrieben, integrieren. Bei erfolgreicher Prüfung wird wie üblich über das Sicherheitsmodul die Prozessaktion als Nachricht signiert (4.), welche dann über das Netzwerkmodul an die externe SCI-Komponente zur Kommunikation an die anderen dpex-Instanzen weitergegeben wird (5.). Wie in Abbildung 40 angedeutet, werden über das `taskAction`-Objekt auch gegebenenfalls die Prozessvariablen übermittelt.

1. `UI:claimTask`
2. `DPEXService::sendTaskAction`
3. `BPM::isConform(model, instance, taskAction)`
4. `SCI.Security::sign(taskAction)`
5. `SCI.Network::sendTaskAction(taskAction)`

Abbildung 47: Aufrufsequenz beim Senden einer Aktion

Empfangen einer Prozessaktion (erste Stufe)

Die Aufrufsequenz in dpex für das Empfangen einer Prozessaktion ist in Abbildung 48 dargestellt. Das dpex-Framework empfängt alle Prozessaktionen, auch die selbst initiierten, als Nachricht über das Nachrichtenmodul in der Methode `handleTaskAction` (6.), welche zum Beispiel als Event-Handler für Smart Contract-Ereignisse registriert ist. Im allgemeinen Fall gibt die Methode die Kontrolle an den ICB (`DPEXService`) ab (7.), welcher zuerst obligatorisch über das Sicherheitsmodul die Signatur der Nachricht überprüft (8.). Anschließend wird im Synchronisierungsmodul basierend auf einem `CandidateLog` die Prozessaktion zwischengespeichert (9.) und der Empfang der Nachricht über eine sogenannte `Acknowledgment`-Nachricht (`ACK`) bestätigt. Nach der Erstellung der `ACK`-Nachricht (10.) wird sie über das Sicherheitsmodul signiert (11.) und über das Netzwerkmodul wieder zur Kommunikation an die SCI-Komponente übergeben (12.). Diese Schritte stellen die erste Stufe dar.

Senden einer ACK-Nachricht

6. `SCI.Network::handleTaskAction(taskAction)`
7. `DPEXService::handleTaskAction(taskAction)`
8. `SCI.Security::verify(taskAction)`
9. `SCI.Consensus::handleTaskAction(taskAction)`
10. `DPEXUtils::buildACK`
11. `SCI.Security::sign(ack)`
12. `SCI.Network::sendACK(ack)`

Abbildung 48: Aufrufsequenz beim Empfangen einer Aktion

Empfangen einer ACK-Nachricht (zweite Stufe)

Die Aufrufsequenz in dpex für das Empfangen einer Bestätigungsnachricht ist in Abbildung 49 dargestellt. Die zweite Stufe beginnt mit

dem Erhalt einer ACK-Nachricht im Netzwerkmodul in der handleAcknowledgment-Methode (13.), welche die ACK-Nachricht an den ICB weiterleitet (14.). Nach Prüfung der Signatur (15.) erlangt das Synchronisierungsmodul die Kontrolle und überprüft nun, ob der Erhalt dieser ACK-Nachricht eine referenzierte Prozessaktion final bestätigt (16.). Dies kann über die Anzahl der erhaltenen ACKs entschieden werden oder auch über einen bestimmten Sender einer ACK-Nachricht. Sofern das Synchronisierungsmodul eine Prozessaktion bestätigt, leitet der DPEXService die Aktion schließlich an die externe BPM-Komponente weiter, welche die Aktion final prüft (17.) und den lokalen Prozessstatus dann endgültig aktualisiert (18.).

Interpretation in BPM-Komponente

```
13. SCI.Network::handleAck
14. DPEXService::handleAck(ack)
15. SCI.Security::verify(ack)
16. SCI.Consensus::isAgreementReached(ack)
17. BPM::isConform(taskAction)
18. BPM::claim(taskAction)
```

Abbildung 49: Aufrufsequenz beim Empfangen einer Bestätigungsnachricht

SONDERFALL ETHEREUM Die Aufrufsequenz in dpex beim Einsatz von Ethereum als SCI ist in Abbildung 50 dargestellt. Wie oben angedeutet, kann der Ablauf bei der Verwendung von Ethereum vereinfacht werden, da keine manuelle Überprüfung der Endgültigkeit einer eingehenden Nachricht erforderlich ist. Zumindest bei der Verwendung des Proof of Authority-Protokolls kann bei einer eingehenden Nachricht, welche über das entsprechende Smart Contract-Ereignis und den handleTaskAction-Event-Handler eingeht, davon ausgegangen werden, dass die korrespondierende Prozessaktion damit final bestätigt ist. Aus diesem Grund wird im Netzwerkmodul des Ethereum-Adapters in der handleTaskAction-Methode ein ACK-Nachrichtenobjekt erstellt und damit direkt die handleAcknowledgement-Methode des DPEXService aufgerufen. Somit wird eine eingehende Nachricht gleichzeitig auch als ACK-Nachricht behandelt. Zusammen mit der speziellen Implementierung des Konsensmoduls, welches beim Aufruf von isAgreementReached standardmäßig den Wert true zurückgibt, erfolgt also eine effiziente Abarbeitung der Nachrichtenkommunikation.

Einstufiges Verfahren bei Ethereum

Automatische, lokale ACK-Erzeugung

Verarbeitung in der BPM-Komponente

6.5.3.7 *Zusammenfassung*

Mit der in diesem Kapitel vorgestellten Beispielimplementierung des konzeptionellen dpex-Framework können Prozesse dezentral ausgeführt werden. Die durchzuführenden Schritte werden hier nochmal

6. `SCI.Network::handleTaskAction(taskAction)`
   - ~~`DPEXService::handleTaskAction(taskAction)`~~
   - ~~`SCI.Security::verify(taskAction)`~~
   - ~~`SCI.Consensus::handleTaskAction(taskAction)`~~
7. `DPEXUtils::buildACK`
   - ~~`SCI.Security::sign(ack)`~~
   - ~~`SCI.Network::sendACK(ack)`~~
   - ~~`SCI.Network::handleAck`~~
8. `DPEXService::handleAck(ack)`
9. `SCI.Security::verify(ack)`
10. `SCI.Consensus::isAgreementReached(ack)`
11. `BPM::isConform(taskAction)`
12. `BPM::claim(taskAction)`

Abbildung 50: Aufrufsequenz beim Einsatz von Ethereum

zusammengefasst. Als Vorbereitung einigen sich die Kollaborationspartner auf ein gemeinsames Prozessmodell, das dazugehörige Organisationsmodell sowie einen Algorithmus oder System als SCI. Falls nötig müssen dafür die Adapter für das dpex-Framework implementiert werden. Für die Anbindung von Camunda und Ethereum sind die Adapter bereits in der `dpex-lib` implementiert, sodass direkt mit dem Starten der dpex-Anwendung begonnen werden kann. Dies kann einfach über das `dpex-build`-Projekt und Docker bewerkstelligt werden. Die Bereitstellung der externen BPM- und SCI-Systeme (zum Beispiel Camunda und Ethereum) wird vorausgesetzt und hier nicht explizit angeführt. Im Anschluss muss gegebenenfalls das externe SCI-System konfiguriert werden. Mit dem oben vorgestellten Beispielkonzept muss ein einziger kollaborationsweiter `DpexSci`-Smart Contract bereitgestellt werden. Dies wird von einem Teilnehmer über die dpex-Anwendung initiiert und die daraus resultierende `DpexSci`-Smart Contract-Adresse wird im Anschluss an die übrigen Kollaborationsteilnehmer übermittelt. Jeder Teilnehmer erstellt dann in der lokal laufenden dpex-Instanz die benötigten Adapterfabriken, welche mit den in dieser Kollaboration zu verwendeten Modellen und Konfigurationsparametern (lokale `camundaUrl` und lokale `sciConnection`) konfiguriert werden, sodass die externen Systeme an das nun zu erstellende `Alliance`-Objekt angebunden werden können. Zur Erstellung einer `Alliance` werden in der dpex-Anwendung also die gewünschten Adapterfabriken ausgewählt. Über die `create`-Methode wird jeweils eine Instanz eines `ProcessModel-`, `BPMEngine-` und `Sci-Adapters` erstellt. Insbesondere wird im Rahmen der Instanziierung

des Ethereum-Adapters bei der Instanziierung des Ethereum-Netzwerkmoduls die Adresse des `DpexSci`-Smart Contracts benötigt, um die `handleInstantiation`-Methode als Event-Handler auf die ausgesendeten Events des Smart Contracts registrieren zu können. Wenn schließlich eine Instanziierungstransaktion verifiziert wird, führt jeder Teilnehmer in dem lokalen Ethereum-Client die entsprechende `DpexSci`-Smart Contract-Funktion aus, welche einerseits einen neuen `DpexProcessInstance` in der Blockchain anlegt und andererseits ein Instanziierungsereignis sendet, welches schließlich den Aufruf der Methode `handleInstantiation` in der `dpex`-Anwendung zur Folge hat. Darin wird ein zweiter Event-Handler registriert, nämlich die Methode `handleTaskAction` auf Events des `DpexProcessInstance`-Smart Contract.

6.6 EVALUATION

Dieses Kapitel stellt die Ergebnisse des Evaluationsschritts aus dem zweiten DSR-Zyklus (Kapitel 6) vor. Dieser zweite Zyklus wird durch die offenen Herausforderungen des ersten DSR-Zyklus (Kapitel 5) motiviert.

Ziel des DSR-Artefakts

Nach Peffers et al. [138] wird im Evaluationsschritt eines DSR-Zyklus geprüft, ob mit dem entwickelten Artefakt die definierten Ziele erreicht werden, um die ursprünglich identifizierten Probleme damit zu lösen. Das Artefakt dieses DSR-Zyklus ist das konzeptionelle Schichtenmodell, welches durch einen Middleware-basierten Ansatz eine strikte Trennung der BPM- und SCI-Domäne bei der dezentralen Ausführung von Prozessen verfolgt. Insbesondere ist anzumerken, dass das Artefakt nicht direkt für eine Prozessausführung konzipiert ist. Stattdessen definiert es einen Rahmen, nach dem Anwendungen zur dezentralen Prozessausführung entwickelt werden können. Dies ist im Demonstrationsschritt gezeigt, worin basierend auf dem vorgestellten Middleware-basierten Ansatz zuerst eine Bibliothek implementiert wird. Diese Bibliothek wird daraufhin in eine Spring Boot-Anwendung integriert, um dadurch eine Middleware nach dem Vorbild des entwickelten Artefakts zu schaffen.

Die Evaluation des Artefakts in diesem DSR-Zyklus erfolgt demnach vordergründig über eine Machbarkeitsstudie und der Implementierung eines Prototyps. Der Prototyp ist bereits detailliert im Kapitel 6.5 beschrieben. In Kapitel 6.6.1 soll schließlich überprüft werden, ob das Artefakt mittels des implementierten Prototyps die in Kapitel 6.3 postulierten Anforderungen umsetzen kann. Es sei nochmals darauf hingewiesen, dass der Prototyp selbst nicht evaluiert wird. Er dient lediglich der Demonstration des Artefakts, wobei das Artefakt auf verschiedenste Weise und in verschiedensten Programmiersprachen implementiert werden kann.

Des Weiteren wird in dem Evaluationsschritt in Kapitel 6.6.2 die Funktionsweise des Frameworks der Funktionsweise der wichtigsten verwandten Arbeiten der Blockchain-basierten Prozessausführung gegenübergestellt.

6.6.1 *Evaluation der Anforderungen*

Dieses Kapitel evaluiert, inwiefern das dpex-Framework die definierten Anforderungen umsetzen kann. Die einzelnen folgenden Abschnitte referenzieren jeweils die Anforderungen aus Kapitel 6.3.

Flexibilität durch Modularisierung

Unterstützung verschiedener Modellierungssprachen

A1: FLEXIBILITÄT HINSICHTLICH PROZESSMODELLIERUNGSSPRACHEN Die Flexibilität hinsichtlich Prozessmodellierungssprachen wird in erster Linie durch die Modularisierung erreicht und wird somit auf konzeptioneller Ebene im dpex-Framework unterstützt. Die Implementierung der Prozessausführung erfolgt unabhängig vom Rest der Architektur und die Module werden über Konnektoren an die Middleware angebunden. Somit können verschiedene Prozessmodellierungssprachen zusammen mit einer kompatiblen Ausführungsmaschine über die Implementierung spezieller Konnektoren in dpex verwendet werden.

In der Demonstration und der aktuellen Implementierung werden ausschließlich BPMN-Prozessdiagramme unterstützt. Allerdings werden für das Prozessmodell per se keine Einschränkungen definiert. Das Prozessmodell in der Demonstration wird beispielsweise lediglich als Bytearray spezifiziert und ähnlich flexibel wird für das Organisationsmodell eine Zeichenkette als Datentyp spezifiziert. Die BPM-Schnittstelle im dpex-Framework ist darüber hinaus agnostisch von BPMN, anderen Modellierungssprachen oder anderen Ausführungsparadigmen. Es werden lediglich abstrakte Konstrukte vorausgesetzt wie der Aktivitätslebenszyklus (`claimTask` oder `completeTask`) oder allgemeine Funktionalitäten der Prozessausführungsdisziplin (`isConform` oder `getWorklist`). Darüber hinaus ist die BPM-Schnittstelle selbst ebenfalls nicht fest definiert und kann hinsichtlich Anforderungen zur Integration weiterer Modellierungssprachen und Ausführungssysteme bei Bedarf flexibel angepasst werden.

A2: UNTERSTÜTZUNG GÄNGIGER WFMS-FUNKTIONALITÄTEN Die zentralisierten WFMSs für die innerbetriebliche Prozessausführung stellen bestimmte Funktionalitäten zur Verfügung, welche von den ersten Blockchain-basierten Ansätzen nicht oder nur zum Teil unterstützt werden. Dazu zählen die Überwachung des Prozessstatus sowie das Bereitstellen sowohl von personalisierten Arbeitslisten als auch von Ereignisprotokollen zu Analysezwecken.

Überwachung

Die Überwachung des globalen Prozessstatus wird auf konzeptioneller Ebene durch verschiedene Punkte umgesetzt. Erstens integriert

das dpex-Framework ein Prozessmodellartefakt, welches die global auszuführenden Schritte enthält. Zweitens wird durch die Kommunikation und Synchronisierung aller Ereignisse von Prozessteilnehmern ein dezentral verwalteter aber global einheitlicher Datenbestand ermöglicht. Drittens definiert die `dpex-lib`-Middleware-Bibliothek eine Hilfsdatenstruktur `EventLog`, worin alle Ereignisse verwaltet werden. Dadurch kann jede dpex-Instanz den Prozessfortschritt jederzeit ableiten. In der aktuellen Version des `dpex-react`-Frontends werden darüber alle erledigten Aufgaben mit den bpmIds des ausführenden Agenten gekennzeichnet und alle offenen Aufgaben werden mit einem blauen Punkt markiert. Alternativ können gegebenenfalls auch über die bestehende Funktionalität eines extern angebundenen WFMS fortgeschrittene Visualisierungen der Überwachung genutzt werden.

personalisierte Arbeitslisten

Bei der Umsetzung der Bereitstellung von personalisierten Arbeitslisten sind zwei Punkte zu nennen. Zum einen ist gezeigt, dass eine flexible Integration der organisationalen Perspektive zur Definition von Ausführungsrechten unterstützt wird, was als Grundlage für die Erstellung von Arbeitslisten dient. Ein zweiter wichtiger Aspekt ist die Implementierung des Lebenszyklus von Aktivitäten, was durch die Aufnahme der Methoden `claim` oder `complete` gewährleistet wird. Dadurch können berechtigte Benutzer den Status von Aktivitäten modifizieren, welche über die persönliche Arbeitsliste mit `getWorklist` abgerufen werden können.

Ereignisprotokoll

Zuletzt wird das Ereignisprotokoll für weitere Analysen analog zur Überwachung entweder direkt über das externe WFMS oder manuell über die `EventLog`-Hilfsstruktur zur Verfügung gestellt.

A3: UNTERSTÜTZUNG GÄNGIGER PROZESSPERSPEKTIVEN Nach Kapitel 2.5 zählen die funktionale, die verhaltensorientierte, die organisationale, die informationsorientierte sowie die operationale Perspektive zu den gängigen Prozessperspektiven. Hinsichtlich der ersten beiden Perspektiven können zumindest die grundlegenden Konstrukte, welche über BPMN in Camunda umgesetzt werden, durch die Anbindung des WFMS an dpex ebenfalls umgesetzt werden.

funktionale und verhaltensorientierte Perspektive

Organisationale Perspektive

Insbesondere ist auch die Integration der organisationalen Perspektive demonstriert, welche über die OrgEngine-Erweiterung des extern angebundenen Camunda WFMS im BPM-Konnektor implementiert ist. Dadurch können im BPMN-Prozessmodell über Annotationen organisationale Ausführungsberechtigungen definiert werden, indem die Attribute aus einem Organisationsmodell referenziert werden. Somit können flexible Konstrukte in der organisationalen Perspektive in dpex integriert werden, indem die im BPMN-Modell spezifizierten Annotationen in der OrgEngine über eine Funktion implementiert werden. In dieser Funktion können das Prozess- und Organisationsmodell sowie die komplette Ausführungshistorie für die Evaluation der Ausführungsrechte analysiert werden.

Informationsorientierte Perspektive

Die Verarbeitung von Daten in der informationsorientierten Perspektive ist in dieser Arbeit zwar nicht explizit vorgestellt worden, ist in der `dpex-lib` allerdings zumindest rudimentär implementiert, sodass darüber die Umsetzbarkeit innerhalb des dpex-Frameworks evaluiert ist [58]. Hinsichtlich der Modellierung können die Datenobjekte in BPMN dazu verwendet werden, um Datenwerte zu spezifizieren, welche zum Beispiel vom Benutzer für die Erledigung einer bestimmten Aktivität zur Verfügung gestellt werden müssen. Aufgrund der Verwendung von sehr generischen Nachrichtenobjekten in der SCI können diese mit nahezu beliebigen Informationen angereichert werden. In diesem Fall ist die Klasse `TaskAction`, welche von `Message` erbt, um ein Feld `ProcessVariables` erweitert, um die Datenwerte zu kommunizieren.

Operationale Perspektive

Bei der Umsetzung der operationale Perspektive in dpex zeigen sich durch die Anbindung des Camunda WFMS implementierungstechnische Hindernisse. Das heißt, die Herausforderungen sind auf dem Funktionsumfang von Camunda begründet und resultieren nicht aus der konzeptionellen Architektur von dpex. Camunda ist für eine zentralisierte Prozessausführung entwickelt, während durch die dezentrale Ausführung bei der Integration in dpex die Aktivitäten auf verschiedenen Knoten unabhängig voneinander und insgesamt mehrfach abgearbeitet werden. Mit den von Camunda zur Verfügung gestellten Bordmitteln lassen sich daher manche Anwendungsfälle in der operationalen Perspektive ohne zusätzlichen Implementierungsaufwand nicht umsetzen. In der operationalen Perspektive werden über Aktivitäten externe Schnittstellen aufgerufen oder bestimmter Quelltext ausgeführt. Im Folgenden werden drei Fälle solcher Aktivitäten im Kontext der zwischenbetrieblichen Prozessausführung beschrieben [167].

Aufruf eines lokalen Systems

Im ersten Fall aktualisiert eine Aktivität ein lokales System außerhalb des Bereichs der Prozessausführung, etwa ein ERP-System. Dabei ist eine mehrfache, jeweils lokale Ausführung der Aktivität das gewünschte Verhalten, sodass die Systeme bei den Kollaborationspartnern jeweils aktualisiert werden. Dabei ist zu beachten, dass die Systeme lokal jeweils möglicherweise über unterschiedliche Endpunkte erreichbar sind. Dies stellt damit bereits eine Herausforderung in der Modellierungsphase dar, denn die aktuelle Version des dpex-Frameworks geht davon aus, dass jeder Teilnehmer exakt das gleiche Prozessmodell implementiert. Da dazu auch die Implementierung der Aktivitäten zählt, ist es zunächst nicht möglich, die Aktivitäten mit unterschiedlichen Endpunkten zu konfigurieren. Eine Lösung wäre eine Entkopplung der auszuführenden Skripte von den operationalen Aktivitäten und vom Prozessmodell und den Einsatz eines *Object Brokers*. Über das Skript kann dann die URL von den Teilnehmern unabhängig voneinander lokal konfiguriert werden [191].

Laden von Daten aus einem externen Dienst

Im zweiten Fall wird von einer Aktivität ein kollaborationsexterner

Dienst aufgerufen, um beispielsweise bestimmte Datenwerte aus externen Systemen auszulesen. Dies ist unproblematisch, wenn der externe Dienst dabei deterministische Ergebnisse wie historische Daten im Flugverkehr oder historische Wetterdaten zurückliefert, da somit jede dpex-Instanz die gleichen Ergebnisse weiterverarbeitet. Komplizierter ist der Fall, wenn der externe Dienst nicht deterministische Ergebnisse wie über die Zeit veränderliche aktuelle Wetterdaten oder Aktienkurse zurückliefert. Durch die dezentrale und dadurch zeitlich leicht verschobene Ausführung können die dpex-Instanzen somit unterschiedliche Ergebnisse erhalten, wodurch die Gefahr von unterschiedlichen Prozessständen entsteht. Als Lösung kann das Abrufen von Daten koordiniert werden und von einem Prozessteilnehmer übernommen werden anstatt einer mehrfachen, dezentralen Ausführung des entsprechenden Prozessschritts. Beim Einsatz von BPMN und Camunda müssen dafür allerdings explizite Schritte für die Koordination eingeführt werden, da Camunda automatisierte Aufgaben direkt beim Eintreffen eines Tokens ausführt. Eine Lösung wird in [191] vorgestellt. Um die Gefahr von Datenmanipulation zu verringern werden bei Blockchain-basierten Systemen zu diesem Zwecke sogenannte *Oracles* eingeführt [127], welche möglicherweise auch in diesem Fall bei der Prozessausführung einen Mehrwert schaffen können.

Aufruf externer Dienste

Im dritten Fall hat das Aufrufen eines externen Dienstes weitere Auswirkungen, etwa das Persistieren von Daten in externen Datenbanken oder das Versenden von E-Mails über einen externen Mailserver. Dabei kann die mehrfache Ausführung einer Aktivität der operationalen Perspektive ungewollte Folgen nach sich ziehen, wenn etwa die Daten redundant abgespeichert werden oder die E-Mail mehrfach versendet wird. Die Herausforderung ist dabei auf die Implementierung mit BPMN und Camunda bezogen, da Service Tasks in Camunda sofort ausgeführt werden, ohne dass die Stufen des Aktivitätslebenszyklus wie `claim`, `start` oder `complete` explizit durchlaufen werden. Konzeptionell besteht eine mögliche Lösung also dadurch, dass auch bei Service Tasks die Ausführungsrechte einem bestimmten Teilnehmer zugeordnet werden müssen, um eine einmalige Ausführung zu koordinieren. Diese Lösung befindet sich aktuell in einer prototypischen Entwicklung [191].

Zusammengefasst sind die Prozessperspektiven damit weitestgehend umgesetzt. Lediglich bei der operationalen Perspektive sind bestimmte Herausforderungen identifiziert. Diese beziehen sich allerdings nicht auf das dpex-Konzept, sondern betreffen die angebundene BPM-Komponente. Die konzeptionelle Herangehensweise in dpex, nämlich die Koordinierung der Ausführung von Service Tasks durch den Aktivitätslebenszyklus, kann mit den Camunda-Bordmitteln nicht umgesetzt werden. Allerdings können ersten Ergebnissen zufolge die

Herausforderungen über eine Anpassung des BPM-Konnektors für BPMN-Prozesse und Camunda dennoch gelöst werden [191].

A4: VERWALTUNG MEHRERER KOLLABORATIONEN Aufgrund divergierender Anforderungen existieren verschiedene Lösungen zur dezentralen (Blockchain-basierten) Prozessausführung. Dies resultiert in einer heterogenen Systemlandschaft für Unternehmen, welche an mehreren Kollaborationen beteiligt sind. Dem entgegen sollte eine holistische Lösung entwickelt werden, welche eine homogene Verwaltung von dezentral ausgeführten Prozessen erlaubt.

Verwaltung mehrerer Kollaborationen über Alliance-Objekte

Bei der Entwicklung von `dpex` wurde der einheitlichen Verwaltung mehrerer Kollaborationen eine hohe Priorität zugewiesen. Die Umsetzung erfolgt durch ein zentrales `Alliance`-Objekt, worüber verschiedene Kollaborationen angelegt und konfiguriert werden können. Auch hinsichtlich der SCI beim Konzept der Anbindung von Ethereum wird jede `Alliance` oder Kollaboration über einen eigenen Smart Contract verwaltet. Das `dpex-react`-Frontend bietet dabei eine einheitliche Oberfläche, worüber die verschiedenen Kollaborationen verwaltet werden können und Nutzer kollaborationsübergreifende Arbeitslisten zur Verfügung gestellt bekommen. Zuletzt ermöglicht die Implementierung des `dpex`-Frameworks durch das Fabrik-Entwurfsmuster die Wiederverwendung von Konnektoren für mehrere Kollaborationen.

Wiederverwendung von Konnektoren

B1: KOMMUNIKATION Die Kommunikation ist im `dpex`-Framework durch das Kommunikationsmodul in der SCI-Komponente repräsentiert. In der Implementierung eines SCI-Konnektors wird für die Anbindung die Klasse `BaseNetwork` erweitert. Die Kommunikation kann dabei entweder über das automatische Versenden von Nachrichten mittels der externen SCI-Komponente umgesetzt werden (`EthereumNetwork`) oder über eine manuelle Implementierung mittels des HTTP-Protokolls, worüber die `dpex`-Instanzen der Kollaborationsteilnehmer aufgerufen werden können.

Implementierung über SCI

manuelle Implementierung

B2: SYNCHRONISIERUNG Analog zur Kommunikation integriert das `dpex`-Framework ein dediziertes Modul für die Synchronisierung. Allerdings wird dieses Modul in der aktuellen Implementierung lediglich durch eine Attrappenimplementierung umgesetzt. Das ist durch das Konzept zur Anbindung der Ethereum-Blockchain begründet, wonach das `dpex`-Framework bereits mit synchronisierten Nachrichten aufgerufen wird. Anders ausgedrückt, die Synchronisierung wird durch die externe SCI-Komponente bereits implementiert.

Warten auf Blöcke bei probabilistischer Finalität

Eine manuelle Implementierung der Synchronisierung ist hingegen notwendig, wenn das Blockchain-Protokoll mit einem Konsensmechanismus mit probabilistischer Finalität konfiguriert ist. Bei Proof of Work etwa können nämlich die bereits *synchronisierten* Nachrich-

ten noch aufgrund eines Blockchain-Forks revidiert werden. In diesem Fall können eingehende Nachrichten in der `ConsensusLog`-Hilfsdatenstruktur vorerst zwischengespeichert werden. Mit zunehmender Anzahl an neuen Blöcken verringert sich die Wahrscheinlichkeit für einen Blockchain-Fork. Das heißt nach einer benutzerkonfigurierten Anzahl an neuen Blöcken kann der SCI-Konnektor die Nachricht als synchronisiert betrachten und an den ICB weiterleiten. Eine Erweiterung der dpex-Implementierung wird aktuell prototypisch entwickelt [81].

manuelle SCI-Implementierung

Eine komplett manuelle Implementierung ist aktuell nicht vorgesehen, weswegen das Synchronisierungsmodul auch bislang nicht entsprechend entwickelt ist. Zum Beispiel verwaltet die `CandidateLog`-Hilfsstruktur aktuell keinen Prozessstatus, das heißt, wenn aktuell zwei verschiedene Teilnehmer die Ausführungsrechte für eine Aktivität beanspruchen würden, würde das `CandidateLog` für beide Nachrichten eine `ACK`-Nachricht senden. Das erwartete Verhalten wäre allerdings, dass nur die erste Anfrage bestätigt wird. Aufgrund der Komplexität schlägt das dpex-Framework vor, eine externe SCI-Komponente wie Ethereum oder BFT-SMaRt [12] für diese Zwecke anzubinden. Ein Konnektor für die Anbindung von BFT-SMaRt wird ebenfalls aktuell entwickelt [81].

B3: SICHERHEIT Das dpex-Framework integriert ein Sicherheitsmodul für die Implementierung bestimmter Sicherheitsbelange. Dieses spezifiziert die Funktionalität zum Signieren von Nachrichten und Verifizieren von Signaturen, sodass hierüber digitale Signaturen in dpex verwendet werden können. Ähnlich zur Synchronisierung wird diese Funktionalität in der aktuellen Implementierung an die Ethereum-SCI ausgelagert. Insbesondere sorgt Ethereum durch digitale Signaturen für die Authentifizierung von Benutzern, sodass die getätigten Aktionen nicht geleugnet werden können. Die aktuelle Implementierung verwendet den `ClientTransactionManager`[21] im `EthereumNetwork` zum Übergeben von Transaktionen an den externen Ethereum-Client. Dies fördert zwar eine unkomplizierte Demonstration von dpex, allerdings erfolgt die Signierung der Nachrichten dabei im externen Blockchain-Knoten, anstatt dafür die privaten Schlüssel der Benutzer zu verwenden. Im produktiven Betrieb ist aus Sicherheitsgründen die Verwendung des `RawTransactionManagers`[22] zur direkten Signierung der Transaktionen in dpex vorzuziehen.

digitale Signaturen

Implementierung über SCI

Verwendung des RawTransaction-Managers

C1: FLEXIBILITÄT Das konzeptionelle dpex-Framework forciert eine strikte Trennung zwischen der BPM- und der SCI-Domäne, welche unter anderem durch die Schnittstellenintegration umgesetzt wird.

Schichtentrennung

21 https://github.com/web3j/web3j/blob/master/core/src/main/java/org/web3j/tx/ClientTransactionManager.java, besucht am 16.01.2024

22 https://github.com/web3j/web3j/blob/master/core/src/main/java/org/web3j/tx/RawTransactionManager.java, besucht am 16.01.2024

flexible Konfiguration der Konnektoren

Bei der Demonstration ist mittels der Implementierung von Konnektoren die Benutzung dieser Schnittstellen gezeigt. In der aktuellen Version von `dpex-lib` stehen im `impl`-Paket verschiedene Konnektoren zur Verfügung: ein `CamundaAdapter` und ein `jBPMAdapter` in der BPM-Domäne sowie ein `EthereumAdapter` und ein `HyperledgerAdapter` in der SCI-Domäne. Zusätzlich wird aktuell ein `BFTSmartAdapter` entwickelt, um neben den Blockchain-Protokollen eine traditionellere Form einer SCI-Umsetzung zu demonstrieren [81]. Über die Konfiguration des zentralen `Alliance`-Objekts und die Koordination über den ICB, welcher auf der höheren Ebene der Schnittstellenbeschreibung arbeitet, können diese Konnektoren unabhängig voneinander ausgetauscht und verwendet werden. Aus Sicht des dpex-Artefakts im Kontext dieses DSR-Projekts können die SCI-Konnektoren gleichermaßen angebunden werden. In weiteren praxisnäheren Studien ist zu evaluieren, wie sich die unterschiedlichen Eigenschaften der SCIs auf die dezentrale Prozessausführung auswirkt (Kapitel 9.3). Gegebenenfalls können für bestimmte Anwendungsfälle geeignete Konnektoren vorgeschlagen werden. Dies wird allerdings nicht im Rahmen dieser Arbeit betrachtet.

weitere Flexibilitätsdefinitionen

In den verwandten Arbeiten (Kapitel 8) werden zudem alternative Definitionen des Flexibilitätsbegriffs verwendet. Die Laufzeitkonfiguration von Prozessen wird zum Beispiel von López-Pintado et al. durch die dynamische Auswahl von Subprozessen umgesetzt [109]. Ähnlich können im Konzept von Zhang et al. Prozessmodellfragmente definiert werden, welche erst zur Laufzeit zusammengesetzt werden [197]. In dieser Hinsicht existieren im dpex-Framework aktuell keine Konzepte. Die Flexibilität in der SCI-Domäne wird unter anderem von Falazi et al. behandelt [46]: Im BlockME2-Ansatz können innerhalb einer einzigen Prozessinstanz verschiedene Blockchain-Protokolle angebunden werden. Allerdings fokussiert sich dieser Ansatz hauptsächlich auf die operationale Perspektive, indem während der Prozessausführung automatisiert Funktionen von Smart Contracts aufgerufen werden statt auf eine Arbeitslisten-basierte Koordination während der Prozessausführung. Der Aufruf von Smart Contract-Funktionen verschiedener Blockchains ist prinzipiell auch in dpex über die operationale Perspektive möglich, wohingegen die Kommunikation und Synchronisierung lediglich über eine einzige SCI abläuft.

Anbindung externer Komponenten

Erweiterung der Komponenten über Adapter

C2: ERWEITERBARKEIT Ein Vorteil des dpex-Frameworks ist die Möglichkeit der Anbindung externer BPM- und SCI-Systeme statt deren manueller (Re)-Implementierung. Die Anbindung wird über die Implementierung von Konnektoren umgesetzt. Dabei ist in dieser Arbeit auch die Erweiterbarkeit von dpex gezeigt, wodurch die extern bereitgestellte Funktionalität um weitere Aspekte ergänzt wird. Insbesondere wird eine Erweiterung von Camunda hinsichtlich der organisationalen Perspektive durch den Einsatz von BPMN-Annotationen

und eine Implementierung mittels der OrgEngine demonstriert. Außerdem wird im `EthereumAdapter` die Herausforderung der probabilistischen Finalität von Proof of Work-Blockchains adressiert und damit eine Erweiterung in der SCI-Domäne umgesetzt.

6.6.2 *Vergleich mit Blockchain-basierten Lösungen*

Das dpex-Framework ist durch die existierenden Probleme von rein Blockchain-basierten Lösungen motiviert, wozu auch das in Kapitel 5 vorgestellte Artefakt zählt. Während Kapitel 6.6.1 das Framework gegen die definierten Anforderungen evaluiert, stellt der folgende Abschnitt das dpex-Framework und Blockchain-basierte Lösungen konzeptionell gegenüber. Dabei fokussiert sich Kapitel 6.6.2.1 darauf, wie jeweils die Mechanismen der Blockchain-Technologie zur Bereitstellung einer sicheren Infrastruktur integriert werden. Darauf aufbauend beschreibt Kapitel 6.6.2.2 kurz das daraus resultierende Problem der Finalität bei dpex.

6.6.2.1 *Konzeptuelle Verwendung der Blockchain*

Sowohl Blockchain-basierte Ansätze als auch dpex, sofern mit einem Ethereum-Adapter konfiguriert, empfangen neue Nachrichten beziehungsweise Ereignisse über (unverifizierte) Transaktionen im lokalen Ethereum-Client. Der Ethereum-Client etwa prüft eingehende Transaktionen durch eine simulierte Ausführung der referenzierten Smart Contract-Funktion (Abbildung 44). An dieser Stelle unterscheiden sich die Blockchain-basierten Ansätze von dpex.

Fat Contract

Bei der Blockchain-basierten Prozessausführung verwaltet der Smart Contract sowohl eine Repräsentation des Prozessmodells als auch den aktuellen Ausführungsstand. Die Transaktionen beschreiben Prozessaktionen, welche über die Ausführung der referenzierten Funktion den Prozessstatus im Smart Contract aktualisieren sollen. Bei erfolgreicher Simulation wird die Transaktion in den Mining-Pool aufgenommen und früher oder später durch die Inklusion in einem Block verifiziert. Sobald ein neuer Block ankommt, welcher die Transaktion verifiziert, wird erneut die von der Transaktion referenzierte Smart Contract-Funktion ausgeführt und dabei die lokale Kopie der Blockchain aktualisiert. Im Zuge dessen wird der Prozessstatus aktualisiert. Wenn die Transaktion eine nicht prozesskonforme Aktion beschreibt, schlägt die simulierte Ausführung im Smart Contract fehl und die Transaktion kann nicht verifiziert und damit auch nicht in die Blockchain integriert werden.

Thin Contract

In dpex repräsentieren die Transaktionen aus technischer Sicht ebenfalls Aufrufe von Smart Contract-Funktionen und beschreiben aus inhaltlicher Sicht die Prozessaktionen. Die Transaktionen durchlaufen somit analog die Schritte der Überprüfung (simulierte Ausführung der Smart Contract-Funktion), die Aufnahme in den Mining-Pool,

die globale Verifikation durch die Integration in einen neuen Block sowie die letztendliche Ausführung mit Auswirkung auf die lokale Blockchain-Kopie, wenn sie in einen neuen Block aufgenommen wird. Der Unterschied zur Blockchain-basierten Prozessausführung besteht darin, dass der Smart Contract weder den Prozessstatus verwaltet noch eine Transaktion auf Konformität prüfen kann. Stattdessen wird die Transaktion und damit die Prozessaktion selbst lediglich in einer Liste abgespeichert. Die entsprechende Smart Contract-Funktion wird somit prinzipiell immer erfolgreich ausgeführt, sodass alle Transaktionen, ob prozesskonform oder nicht, verifiziert und gespeichert werden. Im dpex-Framework übernimmt die Ethereum-Blockchain wie beschrieben lediglich die Aufgaben der Kommunikation, Synchronisierung und Sicherheit. Die Ausführung von Prozessen oder die Prüfung von Konformität ist explizit ausgeschlossen und wird von den BPM-Komponenten übernommen. Das heißt, dass Prozessaktionen nicht direkt über das Ethereum-Protokoll in den Ethereum-Clients oder in den Smart Contracts verifiziert werden, sondern in lokalen Ethereum-externen Komponenten. Dabei ist entscheidend, dass für alle Teilnehmer die gleiche Reihenfolge der Nachrichten sichergestellt wird, was im Synchronisierungsmodul von dpex, genauer gesagt durch die global eindeutige Ordnung der Transaktionen in der Ethereum-Blockchain umgesetzt wird. Der Unterschied besteht also einerseits in der Auswahl der Transaktionen, welche auf der Blockchain gespeichert werden: ausschließlich *konforme* Transaktionen (Blockchain-basiert) beziehungsweise alle Transaktionen (dpex). Andererseits wird die Prozesskonformität an unterschiedlicher Stelle geprüft: im Ethereum-Client direkt (Blockchain-basiert) beziehungsweise in einer externen BPM-Komponente (dpex).

Integration aller Transaktionen in die Blockchain

dpex versus Blockchain-basiert

Beim Einsatz von Thin Contracts in dpex können damit die gleichen Informationen extrahiert werden wie bei den Fat Contracts der Blockchain-basierten Ansätze. Darüber hinaus werden bei Thin Contracts auch Transaktionen in der Blockchain gespeichert, welche nicht prozesskonforme Aktionen beschreiben. Bei bei Fat Contracts werden diese hingegen nicht berücksichtigt.

6.6.2.2 *Probabilistische Finalität in dpex*

Blockchain-Fork

Wenn bei einem Konsensverfahren wie Proof of Work zwei valide Blöcke annähernd zeitgleich erstellt und propagiert werden, spricht man von einem Blockchain-Fork. Dadurch kommt es zu einer zeitweisen Partitionierung des Blockchain-Netzwerks [190, Kap. 2.2]. Die zwei Partitionen nehmen jeweils einen verschiedenen Block als den aktuell gültigen an und suchen darauf aufbauend weitere Blöcke. Üblicherweise integriert ein Blockchain-Protokoll eine Regel, welche Kette im Falle eines Forks ihre Gültigkeit behält. Für das konkrete Auswahlverfahren wurde in Ethereum kürzlich eine sogenannte *LMD Ghost*-Regel eingeführt [190]. Zur Auflösung eines Blockchain-Forks

Auflösen des Forks

aktualisieren alle Knoten ihre lokale Kopie der Blockchain und behalten die nach Anwendung der Regel einzig gültige Version. Durch diesen Vorgang können Transaktionen, welche vorerst durch einen Block validiert waren, aus der Blockchain gelöscht werden, weil sie in der gültigen Blockchain nicht validiert waren. Bei Ansätzen der Blockchain-basierten Prozessausführung wird dabei direkt der aktuell gültige Prozessstatus verändert. Dies hat den Nachteil, dass die Routine zum Auflösen der durch die Wettlaufsituation entstandenen Netzwerkpartitionierung direkten Einfluss auf die BPM-Domäne hat. Bereits ausgeführte Aktivitäten können plötzlich zurückgesetzt und wieder in einer Arbeitsliste angeboten werden, während automatisiert ausgeführte Aktivitäten erneut ausgeführt werden. Durch die Entkopplung von BPM und SCI in dpex hat ein Blockchain-Fork und dessen Auflösung zuerst einmal keinen Einfluss auf den in den BPM-Komponenten verwalteten Prozessstatus. Allerdings kann sich dadurch die in den Smart Contracts gespeicherte Liste an Ereignissen verändern, wodurch eine Diskrepanz in der global akzeptierten Ereignisliste und dem lokal verwalteten Prozessstatus entstehen kann. Dadurch kann es zu unterschiedlichen Auffassungen des aktuellen Prozessstaus kommen, insbesondere von dpex-Instanzen, welche zuvor unterschiedliche Blockchain-Versionen als die gültige Version angesehen haben. Zur Auflösung des Konflikts müsste der Prozessstatus in den BPM-Komponenten zurückgesetzt werden und auf die neue, global akzeptierte Liste an Ereignissen angepasst werden. Da dies unter Umständen von den angebundenen WFMSs nicht unterstützt wird, schlägt dpex vor, auf eine konfigurierbare Anzahl von Blöcken zu warten.

Auswirkungen aus BPM-Sicht

Teil IV

SYNTHESE

7

EVALUATION

In Kapitel 5 und Kapitel 6 werden im Rahmen eines DSR-Projekts zwei Artefakte im Bereich der dezentralen Prozessausführung vorgestellt. Der Kern dieser Forschungsarbeit wird durch die zwei DSR-Zyklen repräsentiert, worin in Kapitel 5.6 und in Kapitel 6.6 die produzierten Artefakte jeweils deskriptiv evaluiert werden. Dieses Kapitel ordnet die Artefakte schließlich in die Gesamtarbeit ein, indem eine finale Bewertung erfolgt und der Bogen zu den ursprünglich gestellten Forschungsfragen geschlossen wird.

Forschungsfrage RQ1

- Wie kann eine interpretierende Ausführung von Geschäftsprozessen, welche über BPMN-Prozessmodelle [135] definiert sind, auf der Ethereum-Blockchain umgesetzt werden?

Diese Forschungsfrage ist mit dem in Kapitel 5 entwickelten Artefakt beantwortet. Über den darin vorgestellten Smart Contract lassen sich ähnlich einem WFMS Prozesse interpretieren und ausführen, wobei kleinere Unterschiede im Vergleich zu beispielsweise der zentralisierten Prozessausführung mit Camunda festzustellen sind.

Blockchain-Anwendung als WFMS

Der Smart Contract definiert Regeln, nach welchen Transaktionen, die bestimmte Prozessaktionen repräsentieren, im Ethereum-Netzwerk als gültig validiert werden und welche Transaktionen ungültig sind. Dafür definiert der Smart Contract einerseits Funktionen und andererseits über Variablen auch eine Datenstruktur, welche innerhalb der lokalen Client-Anwendung ausgeführt und gespeichert werden. In diesem Sinne übernimmt die Client-Anwendung die Rolle des WFMS.

Unterschiede zwischen Artefakt und Camunda

Während bei Camunda zuerst ein Prozessmodell bereitgestellt wird, bevor dieses über die Benutzerschnittstelle instanziiert werden kann, ist in dem präsentierten Artefakt kein Repositorium für Prozessmodelle vorgesehen. Stattdessen wird über die Bereitstellung des Smart Contracts durch dessen Adresse ein bestimmter Speicherbereich reserviert, in dem der Prozessstatus einer einzigen Prozessinstanz verwaltet wird. Anschließend wird über Transaktionen das Prozessmodell in diesem abgeschlossenen Speicherbereich bereitgestellt. Schließlich kann über Transaktionen der Prozessstatus aktualisiert werden,

wobei ausschließlich valide, also der Prozessmodellsemantik entsprechende Transaktionen berücksichtigt werden.

Der Smart Contract kapselt sowohl das Prozessmodell, die Verwaltung einer einzigen Prozessinstanz als auch die Implementierung der Semantik, das heißt, die Prüfung, ob eine bestimmte Prozessaktion konform ist oder nicht. In diesem Sinne existieren Diskrepanzen zwischen der Architektur der Prozessausführung mit diesem Artefakt und der zentralisierten Prozessausführung in einem traditionellen WFMS. Für eine ähnlichere Umsetzung könnte die Semantik in einem separaten Smart Contract als *Workflow Engine* bereitgestellt werden, ein weiterer Smart Contract wäre für die Verwaltung aller bereitgestellten Prozessmodelle verantwortlich und über die Workflow Engine könnten diese Prozessmodelle instanziiert und ausgeführt werden. Die Umsetzbarkeit und die praktische Anwendbarkeit, insbesondere hinsichtlich Vertraulichkeit von Daten und Kosten, vor allem auf einer öffentlichen Blockchain, bleiben jedoch zu evaluieren.

rudimentäre Umsetzung erfolgt

Zusammenfassend bleibt festzustellen, dass über den Smart Contract jeweils lokal in den Blockchain-Knoten der Prozessstatus verwaltet und geprüft werden kann und somit die grundlegendsten Prinzipien der dezentralen Prozessausführung umgesetzt sind. Weiterhin ist gezeigt, dass sich die organisationale und informationsorientierte Prozessperspektive rudimentär über den vorgestellten Smart Contract umsetzen lassen [168, 169]. Allerdings werden fortgeschrittene WFMS-Funktionalitäten wie personalisierte Arbeitslisten oder eine nachvollziehbare Historie nicht direkt unterstützt, sondern müssen manuell implementiert werden, wobei die benötigten Daten dafür über die Blockchain bereitgestellt werden.

Fazit RQ1. Eine interpretierende Ausführung ist durch das entwickelte Artefakt zwar möglich, allerdings sind einige Funktionen von WFMSs, so wie in Kapitel 2 vorgestellt, darin nicht implementiert. Dennoch ist die ursprüngliche Forschungsfrage beantwortet und das Artefakt erwies sich durch die Berücksichtigung und Weiterentwicklung in weiteren Forschungsarbeiten als wegweisend im Bereich der Blockchain-basierten Prozessausführung.

Forschungsfrage RQ2

- Wie kann ein Artefakt eine flexible dezentrale Prozessausführung umsetzen und dabei unabhängig von bestimmten Modellierungssprachen, Blockchain-Protokollen oder zugrundeliegenden dezentralen Infrastrukturen bleiben?

Alternative zur Blockchain-basierten Prozessausführung

Anstatt das Artefakt für die rein Blockchain-basierte Prozessausführung weiterzuentwickeln, wird mit der zweiten Forschungsfrage die grundsätzliche Architektur infrage gestellt, die bislang verfolgt

wurde. Dabei wird nicht prinzipiell die Relevanz der rein Blockchain-basierten Prozessausführung diskutiert, wofür in der Literatur bereits sinnvolle Anwendungsszenarien evaluiert werden. Vielmehr soll eine alternative Architektur vorgeschlagen werden, um einerseits die aktuell erforderliche Neuimplementierung von WFMS-Funktionalitäten zu vermeiden und andererseits die unkomplizierte Integration verschiedenster Systeme zur dezentralen Verwaltung der Prozesskontrolldaten zu ermöglichen.

Unterschiede der beiden Ansätze

Im Gegensatz zur rein Blockchain-basierten Lösung ist die vorgeschlagene Architektur nicht ohne zusätzliche Blockchain-externe Komponenten funktionsfähig und erfordert die Integration einer BPM-Komponente für die Interpretation der Prozesse. Ebenso bleibt die Architektur zudem unabhängig von spezifischen Modellierungssprachen oder bestimmten Implementierungen von dezentralen Systemen. Zuletzt erfolgt durch die Anbindung externer Komponenten, vor allem innerhalb der BPM-Domäne, grundsätzlich eine entwurfsseitige Unterstützung der Prozessperspektiven innerhalb des Decentralized Control-Prinzips.

dpex als Verallgemeinerung der Blockchain-basierten Prozessausführung

Im Prinzip verallgemeinert das entworfene dpex-Framework die Ansätze der rein Blockchain-basierten Prozessausführung. In beiden Ansätzen werden die Prozesskontrolldaten dezentral verwaltet, indem geeignete Algorithmen die jeweils lokalen Datenstände synchronisieren. Ebenfalls werden gleichfalls alle Prozessaktionen über Nachrichten kommuniziert, welche über die Synchronisierungsalgorithmen global geordnet werden. Der Unterschied zwischen den Ansätzen besteht darin, dass bei der Blockchain-basierten Prozessausführung die Implementierung ausschließlich innerhalb der Client-Anwendung im Rahmen von Smart Contracts erfolgt. Das dpex-Framework hingegen verwendet die Blockchain ebenfalls zur Kommunikation und Synchronisierung von Nachrichten, wobei die Interpretation der korrespondierenden Prozessaktionen in externen Komponenten erfolgt. Dies ermöglicht dem dpex-Framework, auch andere Synchronisierungsalgorithmen anstelle der Blockchain anzubinden.

Zusammenfassend unterstützt das dpex-Framework die Interpretation von BPMN-Prozessdiagrammen und kann durch eine Erweiterung (OrgEngine) auch eine fortschrittliche Interpretation der organisationalen Perspektive umsetzen. Weiterhin ist in [58] gezeigt, wie die informationsorientierte Perspektive durch die dezentrale Verwaltung von Prozessvariablen integriert werden kann. Allerdings sind auch ungelöste Herausforderungen identifiziert. Wie in Kapitel 2.5.4 beschrieben, umfasst die informationsorientierte Perspektive neben Prozessvariablen auch die organisierte Integration von Unternehmensdaten in die Prozessausführung. Inwiefern dies bei zwischenbetrieblichen Prozessen relevant und umsetzbar ist, bleibt in künftigen Arbeiten zu evaluieren (Kapitel 9.3). In dieser Hinsicht können auch dezentrale Daten- und Dokumentenverwaltungsstrategien wie das *InterPla-*

netary File System[1] untersucht werden. Ähnlich bestehen auch noch Herausforderungen in einer fortschrittlichen Umsetzung der operationalen Perspektive. Dies umfasst zu entwickelnde Konzepte für die Anbindung externer Dienste oder die koordinierte Ausführung von externen Programmen, wobei zu unterscheiden ist, ob die Programme ebenfalls in jeder Organisation auszuführen sind oder stattdessen eine einmalige Ausführung gewünscht ist (Kapitel 6.6).

Fazit RQ2. Das Middleware-basierte dpex-Framework erlaubt die Integration von existierenden WFMSs und dezentralen Systemen für flexible Konfigurationsmöglichkeiten zur zwischenbetrieblichen Prozessausführung. Durch Beispielimplementierungen wird demonstriert, wie unter anderem das Camunda WFMS für die Interpretation von Prozessen und die Ethereum-Blockchain als SCI für die dezentrale Verwaltung der Prozesskontrolldaten angebunden werden können. Eine *vollwertige* Integration der Prozessausführung kann dagegen noch nicht attestiert werden, primär aufgrund der eingeschränkten Umsetzung der operationalen Perspektive in Bezug auf die identifizierten Herausforderungen.

Gesamtfazit

Zusammenfassend werden mit dieser Arbeit neue Paradigmen in der dezentralen Prozesskontrolldatenverwaltung vorgestellt. Durch den interpretierenden Ansatz in der Blockchain-basierten Prozessausführung und insbesondere durch das dpex-Framework bietet sich zum einen neben dem traditionelleren Prinzip der nachrichtenbasierten Synchronisierung lokal ausgeführter Prozesse (Loosely Coupled Workflows) eine flexible Alternative zur zwischenbetrieblichen Prozessausführung. Weiterhin erweitert das dpex-Framework die Möglichkeiten in der dezentralen Prozesskontrolldatenverwaltung. Unterschiedliche SCIs und auch nicht Blockchain-basierte Systeme können flexibel mit verschiedenen BPM-Systemen unter Berücksichtigung verschiedener Prozessperspektiven zur dezentralen Prozessausführung integriert werden.

Statt der vor dieser Arbeit zur Verfügung stehenden Möglichkeiten der Loosely Coupled Workflows oder dem kompilierenden Ansatz bei der Blockchain-basierten Prozessausführung, ist nun ein Framework zur multiperspektivischen Prozessausführung mit fortgeschrittenen WFMS-Funktionalitäten, wie der Überwachung oder der Bereitstellung kollaborationsweiter, personalisierter Arbeitslisten basierend auf einem Organisationsdiagramm bei der dezentralen Prozessausführung verfügbar.

1 https://ipfs.tech/, besucht am 07.03.2024

8

RELATED WORK

Dieses Kapitel stellt die verwandten Arbeiten mit Bezug zum Forschungsbereich dieser Dissertation vor. Dafür wird zuerst in Kapitel 8.1 die Strategie für das Auswählen relevanter Veröffentlichungen beschrieben und die vollständige Liste der selektierten Publikationen tabellarisch aufgearbeitet. Im Anschluss werden die wichtigsten Arbeiten der Blockchain-basierten Prozessausführung in Kapitel 8.2 detaillierter betrachtet. Daraufhin werden diese Ansätze in Kapitel 8.3 zu den Artefakten dieser Dissertation genau abgegrenzt und beschrieben, worin die Unterschiede bestehen und wie die bestehende Literatur die Konzepte dieser Dissertation beeinflusst. Während die Kernarbeiten der Blockchain-basierten Prozessausführung näher untersucht werden, stellt Kapitel 8.4 die weiter entfernten Arbeiten, etwa die Bereiche Modellierung oder Überwachung betreffend, auf einer höheren Abstraktionsstufe vor. Kapitel 8.5 fasst die Ergebnisse von bereits bestehenden Literaturanalysen zusammen. Zuletzt beschreiben Kapitel 8.6 und Kapitel 8.7 die übrigen Arbeiten, welche durch das Auswahlverfahren zwar berücksichtigt sind, jedoch nur einen entfernten Bezug zu dieser Dissertation aufweisen.

Es sei an dieser Stelle auf die Diskussion verwandter Arbeiten in Kapitel 3.5 verwiesen, worin die dezentrale Prozesskontrolldatenverwaltung als neue Form der Workflow-Interoperabilität in die zwischenbetriebliche Prozessausführung eingeordnet wird. Das Ziel dieses Kapitels hingegen ist es, die wichtigsten Forschungsergebnisse aus dem Bereich der interdisziplinären Forschung der beiden Domänen zwischenbetriebliche Prozessausführung und Blockchain-Technologie vorzustellen.

8.1 AUSWAHL DER LITERATUR

Die ausgewählten Veröffentlichungen, welche in den folgenden Kapiteln näher vorgestellt werden, sind nach einem vierstufigen Verfahren selektiert. Die ersten drei Phasen orientieren sich an der Schneeball-Methode, eine anerkannte Strategie für systematische Literaturanalysen [188]. Die letzte Phase ergänzt das Verfahren dabei im Sinne einer Qualitätssicherung. Die Phasen werden nun kurz vorgestellt, bevor die einzelnen Artikel beschrieben und analysiert werden. Die

Schneeball-Methode

Nummer hinter der Benennung der Phase deutet im Folgenden auf die jeweilige Anzahl der zur Literaturstudie hinzugefügten Publikationen hin.

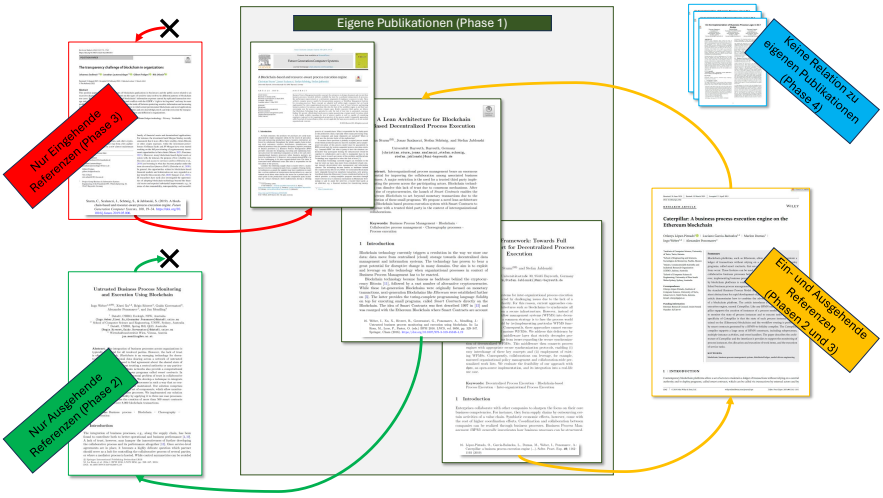

Abbildung 51: Methodik der Literaturrecherche

PHASE 1: STARTMENGE (5) Die erste Phase der Schneeball-Methode besteht darin, wenige, aber relevante Veröffentlichungen für das weitere Vorgehen auszuwählen. Die Startmenge wird hier über die eigenen Publikationen bestimmt. In diesen Publikationen ist bereits eine Vorauswahl der verwandten Arbeiten enthalten, wodurch diese sich als Startmenge auszeichnen. Die Publikationen sind in Abbildung 51 in der grauen Box in der Mitte angedeutet.

PHASE 2: BACKWARD SNOWBALLING (35) Beim *Backward Snowballing* (deutsch: Rückwärtssuche) werden ausgehend von der Startmenge diejenigen Publikationen betrachtet, welche in den jeweiligen relevanten Kapiteln, zum Beispiel *Related Work* (deutsch: verwandte Arbeiten), analysiert werden. Auf eine Auswertung der vollständigen Referenzliste wird verzichtet, um hier nicht relevante Quellen wie Grundlagenliteratur zu kryptografischen Verfahren auszuschließen. In Abbildung 51 wird links unten eine externe Publikation von einer eigenen Publikation referenziert. Die externe Publikation weist dabei keine Referenz auf eine eigene Veröffentlichung auf. Hat die referenzierte externe Publikation dennoch einen Verweis auf mindestens eine eigene Publikation (Abbildung 51, rechts unten), wird sie fortan mit der Farbe orange kodiert.

PHASE 3: FORWARD SNOWBALLING (44) Die dritte Phase, *Forward Snowballing* (deutsch: Vorwärtssuche) besteht darin, wieder basierend auf der Startmenge die Veröffentlichungen zu suchen, welche mindestens eine Publikation aus der Startmenge, also

hier eine eigene Publikation, referenzieren. Auch hier wird analog zur vorherigen Phase die Suche auf relevante Kapitel eingeschränkt. In Abbildung 51 ist links oben eine externe Publikation abgebildet, welche zwar mindestens eine der eigenen Publikationen referenziert, jedoch von keiner der eigenen Publikationen referenziert wird. Für die Suche werden die beiden Suchmaschinen *Research Gate* und *Google Scholar* verwendet, welche zuletzt am 21.04.2024 geprüft wurden.

PHASE 4: BETRACHTUNG VON SLRS (10) In der vierten Phase wird eine bestehende Literaturanalyse verwendet, um die ursprüngliche Einschränkung durch die Startmenge zu prüfen. Basierend auf der Arbeit von Stiehle et al. [164], einer Analyse von Ansätzen zur Prozessausführung (englisch: *enactment*) mithilfe der Blockchain-Technologie, werden der Ergebnisliste weitere Veröffentlichungen hinzugefügt, die durch die bisherigen Phasen nicht erfasst werden. Die geringe Anzahl von vier Veröffentlichungen, welche später als relevant erachtet werden, weist auf eine bereits positive Abdeckungsrate der verwandten Arbeiten hin. Publikationen der vierten Phase sind in Abbildung 51 oben rechts in blau dargestellt.

Aus dieser vorläufigen Liste werden anschließend noch Ergebnisse nach bestimmten Kriterien ausgeschlossen, darunter Abschlussarbeiten oder fremdsprachige, also nicht deutsch- oder englischsprachige Literatur. Durch die gewählte Strategie soll im Folgenden ein breiter Überblick über das Forschungsgebiet ermöglicht werden.

Legende für Tabelle 7

Die finale Ergebnisliste des Auswahlverfahrens ist in Tabelle 7 aufgezeigt. Die Spalten tragen die folgenden Informationen. Die erste Spalte zeigt das Jahr der Veröffentlichung. Die zweite Spalte enthält die Phase, in der die Veröffentlichung der Liste hinzugefügt wurde: Phase 1 (dunkelgrün), Phase 2 (grün beziehungsweise orange), Phase 3 (rot) und Phase 4 (blau). Die dritte Spalte referenziert die Veröffentlichung selbst. In den folgenden Spalten wird die jeweilige Nummer der Referenz in der Publikation angegeben.[1] Referenzen innerhalb der Rückwärtssuche weisen grüne Nummern auf, während Referenzen der Vorwärtssuche in rot gehalten sind. In der letzten Spalte erfolgt eine Kategorisierung der Arbeiten nach Relevanz. Es ist zu beachten, dass die Farbkodierung zwar die gleichen Signalfarben verwendet, jedoch unabhängig von der Kodierung für die Einordnung in die Phasen ist. Die Kategorien unterteilen sich in eigene Publikationen (dunkelgrün), sehr relevanten Arbeiten in Bezug auf die Blockchain-basierte Prozessausführung (grün), Arbeiten mit einem entfernteren Bezug (orange), Arbeiten mit geringem Bezug zum

1 Zum Beispiel ist die Arbeit von Hull et al. [73] in der Publikation von Sturm et al. [170] (Phase 1) an siebter Stelle geführt (erste Zeile in Tabelle 7), während die Arbeit von Xu et al. [192] eine Referenz mit der Nummerierung 17 auf Sturm et al. [170] aufweist (letzte Zeile in Tabelle 7) und so weiter.

Forschungsbereich (rot), systematische Literaturanalysen (blau) und sonstige Zitierungen (farblos).

Jahr	Phase	Veröffentlichung	Zitiert als [170]	[168]	[169]	[166]	[167]	Kategorie
2016	○	Hull et al. [73]	7	43	8			Modellierung
	○	Idelberger et al. [74]			9			Impl. von Smart Contracts
	○	Weber et al. [185]	16	38	22	27	27	Ausführung
2017	○	Garcia-Banuelos et al. [53]	5	39				Ausführung
	○	Lopez-Pintado et al. [110]	9	44				Ausführung
2018	○	Di Ciccio et al. [33]		42	4			Überwachung
	○	Fridgen et al. [50]			7	6		Fallstudie
	○	Madsen et al. [115]	9	45	18		18	Ausführung
	○	Mendling et al. [120]	10	3				Challenge Paper
	○	Mendling et al. [119]		37				Challenge Paper
	○	Mercenne et al. [122]		41				Ausführung
	○	Nakamura et al. [134]			20			Ausführung
	○	Tran et al. [176]			25			Ausführung
2019	☍	Bore et al. [14]						Generierung von GUIs
	○	Falazi et al. [45]					8	Ausführung
	○	Falazi et al. [46]			6			Ausführung
	○	Klinkmüller et al. [86]				9		Analyse
	◐	Ladleif et al. [96]	15		11			Ausführung
	◐	Lopez-Pintado et al. [113]	18		16	12	16	Ausführung
	○	Lopez-Pintado et al. [108]		40	15		15	Ausführung
	◐	Lopez-Pintado et al. [109]	13		17			Ausführung
	☍	Meroni et al. [124]						Überwachung
	○	Mühlberger et al. [128]				13		Analyse
		Sturm et al. [168]	9		23	17	22	Ausführung
		Sturm et al. [170]			24	18	23	Ausführung
	●	Haarmann et al. [62]	13					zeitliche Simulation
	●	Haarmann et al. [64]	16					Entscheidungsfindung
	●	Köpke et al. [88]		21				Sicherheit
	☍	Silva et al. [159]						Fallstudie
2020	◐	Abid et al. [4]	8				1	Ausführung
	◐	Adams et al. [5]	14				2	Ausführung
	○	Alves et al. [6]					3	Ausführung
	☍	Brahem et al. [16]						Ausführung
	●	Chang et al. [24]		32				SLR im Bereich SCM
	●	Corradini et al. [27]	28					Ausführung
	●	Evermann et al. [43]		13*				Ausführung
	◐	Evermann et al. [44]		–			7	Ausführung
	●	Farrugia et al. [48]		–				falsch zitiert
	●	Garcia-Garcia et al. [54]	49	79				SLR
	●	Javed et al. [80]		–				Politische Entscheidungen
	◐	Klinger et al. [84]	14				12	Ausführung
	●	Klinger et al. [85]	6					Ausführung
	●	Köpke et al. [87]		–				kein wiss. Beitrag
	○	Kumar et al. [91]				10		Sicherheit
	◐	Ladleif et al. [94]	14				13	Ausführung
	◐	Ladleif et al. [95]	19				14	Ausführung
	☍	Lichtenstein et al. [104]						Ausführung
	○	Müller et al. [129]				14		Sicherheit
	☍	Nagano et al. [131]						Ausführung
	☍	Nagano et al. [132]						Ausführung
	●	Ndadji et al. [196]	32	35				allgemeine Modellierung
	☍	Osterland et al. [137]						Ausführung
	☍	Prybila et al. [143]						Überwachung
	◐	Schinle et al. [152]		12			20	Ausführung
		Sturm et al. [169]	24	23		16		Ausführung
		Sturm et al. [166]	18	17	16			Sicherheit
	●	Wang et al. [184]		5				falsch zitiert

Jahr	Phase	Veröffentlichung	Zitiert als [170]	[168]	[169]	[166]	[167]	Kategorie
2021	●	Bodziony et al. [13]				62		Impl. einer Wallet
	●	Braun et al. [17]	19					Dokumentation
	◐	Corradini et al. [28]	19				4	Ausführung
	●	Henry et al. [69]		27				SLR
	●	Henry et al. [68]		11				Task Management
	●	Kowalski et al. [89]		–				Interviews bzgl. Sicherheit
	●	Ladleif et al. [93]	101		102			Dissertation
	◐	Loukil et al. [111]	11				17	Ausführung
	☍	Lu et al. [112]						Ausführung
	●	Morales-Sandoval et al. [114]	–					Ausführung
	●	Yu et al. [194]		17	18			fremdsprachig
	●	Zhang et al. [197]	13					Ausführung
2022	●	Amaral de Sousa et al. [160]		15				Modellierung
	●	Corradini et al. [29]	61	60				Ausführung
	●	Di Ciccio et al. [34]	67					Überwachung
	●	Henry et al. [67]		21				Fallstudie
	●	Henry et al. [66]		195				Dissertation
	●	Kaiya et al. [82]			16			korrelierende Aktivitäten
	●	Lopez-Pintado et al. [107]		15				Ausführung
	◐	Naha et al. [175]	25	23	24		24	Ausführung
	●	Sedlmeir et al. [156]		–				Blockchain-Analyse
	◐	Stiehle et al. [164]	25	38			21	SLR
	◐	Viriyasitavat et al. [178]	128	127			26	SLR
	●	Wan et al. [182]		–				falsch zitiert
2023	●	Curty et al. [30]	231	232	233			SLR
	●	Köpke et al. [92]		45				Modellierung
	●	Lichtenstein et al. [103]	28	27				SLR
	●	Rikken et al. [145]		5				DAO Definitionen
	●	Sali et al. [149]		11				SCM
	●	Stiehle et al. [163]	19					Position Paper: Skalierbarkeit
		Sturm et al. [167]	23	22				Ausführung
	●	Suliman et al. [171]	23					Konformitätsprüfung
	●	Swain et al. [172]		4				SCM
	●	Viriyasitavat et al. [177]	77	86				SLR
	●	Wang et al. [183]		17				Ausführung
	●	Xu et al. [193]	38					Zugriffskontrolle
	●	Xu et al. [192]	17					Ausführung

Tabelle 7: Übersicht über alle Arbeiten, welche durch die Literaturrecherche selektiert wurden

Die relevantesten Arbeiten (grün in der rechten Spalte kategorisiert) werden in Kapitel 8.2 näher beschrieben. Anschließend erfolgt in Kapitel 8.3 eine Abgrenzung dieser sehr relevanten Arbeiten zu den in dieser Dissertation vorgestellten Artefakten. Weitere Arbeiten im interdisziplinären Bereich der Domänen Prozessmanagement und Blockchain (orange kategorisiert) werden in Kapitel 8.4 innerhalb der Unterkategorien Challenge Paper, Modellierung, Analyse, Überwachung, Entscheidungsfindung, Vertraulichkeit/Sicherheit und Fallstudien vorgestellt. Darauf folgt ein Überblick über bereits existierende systematische Literaturanalysen (blau kategorisiert) in Kapitel 8.5. Zuletzt werden in Kapitel 8.6 und Kapitel 8.7 die übrigen Arbeiten mit entferntem Bezug zum Forschungsbeitrag (rot kategorisiert) beziehungsweise aussortierte Arbeiten (farblos kategorisiert) zusammengefasst.

8.2 BLOCKCHAIN-BASIERTE PROZESSAUSFÜHRUNG

2016

BPMN Choreografien auf Ethereum

Weber et al.[↑] veröffentlichten 2016 einen der ersten Ansätze, bei dem eine Blockchain aktiv zur Unterstützung zwischenbetrieblicher Prozessausführung eingesetzt wird [185]. Darin wird basierend auf einer BPMN-Choreografie ein Smart Contract generiert, welcher die Konformität von Aktionen prüft. Der Smart Contract kann sowohl zum passiven Überwachen als auch zur aktiven Mediation eingesetzt werden. Dabei rufen die Prozessteilnehmer die Funktionen des Smart Contracts auf, wodurch gegebenenfalls automatisch die jeweils nächsten Prozessteilnehmer gemäß der Choreografie informiert werden.

2017

Optimierte BPMN Prozessdiagramme auf Ethereum

Auf Weber et al. [185] aufbauend entwickelten **García-Bañuelos et al.**[↑] ein optimiertes Verfahren, durch welches Kosten vor allem bei der Instanziierung eines Prozessmodells eingespart werden können [53]. In ihrem Ansatz verwenden die Autoren ein BPMN-Prozessmodell für die Beschreibung der zwischenbetrieblichen Kollaboration. Die Optimierung erfolgt durch die Transformation in ein Petri-Netz, welches in ein Semantik-äquivalentes Petri-Netz reduziert wird, um einen effizienteren Smart Contract zu generieren.

Caterpillar-Demo

Mit CATERPILLAR [110] stellen **López-Pintado et al.**[↑] einen ersten Prototyp vor, mit dem BPMN-Prozessdiagramme auf der Ethereum-Plattform ausgeführt werden können. Bei der Entwicklung verwenden die Autoren unter anderem die oben vorgestellten Ansätze [53, 185].

2018

DCR-Graphen auf Ethereum

Entgegen dem Caterpillar-System unterstützt der Ansatz von **Madsen et al.**[↑] mit DCR-Graphen eine deklarative Prozessmodellierungssprache [115]. Aus Kostengründen schlagen die Autoren ein Konzept basierend auf einem einzigen Smart Contract (englisch: *mono contract*) vor, welcher alle Prozesse verwaltet, anstatt für mehrere Prozesse oder mehrere Instanzen verschiedene Smart Contracts bereitzustellen.

Caterpillar: Rollenkonzept-Erweiterung

Mercenne et al.[↑] erweitern die Caterpillar-Demo um eine Implementierung der organisationalen Perspektive [122]. Hierfür wird Caterpillar um ein Rollen-Konzept ergänzt, welches auf einer Liste von Mitarbeitern und Rollen sowie deren Zuordnung zueinander basiert. Auf technischer Ebene wird für die Implementierung in der XML-Repräsentation des BPMN-Prozessmodells bei den User Tasks ein At-

tribut *roles* hinzugefügt. Ein sogenannter *(Proxy)AccessManager* setzt die entsprechende Logik in der Prozessausführung um.

Interpretation von Prozessen auf Hyperledger

Während die bisher vorgestellten Ansätze auf dem Ethereum-Protokoll basieren, integrieren **Nakamura et al.**[↑] 2018 die Hyperledger-Blockchain in die Prozessausführung [134]. Für die Ausführung werden BPMN-Prozessmodelle in Zustandsautomaten überführt, welche dann mithilfe einer sogenannten *StateChartEngine* interpretiert werden. Eine weitere Funktionalität des Prototyps ist die Generierung von Fragmenten von Web-Applikationen, deren Verhalten den Definitionen der Zustandsautomaten beziehungsweise Prozessmodellen entspricht. Zu bemerken ist, dass hier erstmals statt einem kompilierenden Ansatz ein interpretierender Ansatz verfolgt wird, bei dem der Smart Contract (beziehungsweise Chaincode) selbst unabhängig von einem spezifischen Prozessmodell ist.

Lorikeet: Asset-Management

Ein weiterer Demo-Prototyp wird mit LORIKEET von **Tran et al.**[↑] vorgestellt, welcher das Prozessmanagement mit dem Asset-Management verbindet [176]. Dabei wird die Speicherung von *Non-Fungable Tokens* während der Prozessausführung unterstützt. Non-Fungable Tokens (NFT) auf einer Blockchain sind digitale Repräsentationen von einzigartigen, unterscheidbaren Gegenständen in der realen Welt.[2] Im Gegensatz zu Kryptowährungen oder traditionellen Banknoten, welche jeweils austauschbar sind, beschreiben NFTs ganz allgemein unterscheidbare Dinge, zum Beispiel ein Grundstück, welches durch die geografischen Koordinaten eindeutig bestimmt ist. Für deren Verwaltung wird im BPMN-Modell ein Element für die Definition einer Registrierungsstelle als Smart Contract hinzugefügt. Der Mehrwert der Blockchain-basierten Prozessausführung ist am Beispiel von Getreideverkäufen, welche als Non-Fungable Assets modelliert sind, gezeigt. Die Getreideverkäufe können so als Datenobjekt bei der Prozessausführung erzeugt, bearbeitet und unter Nutzung der Sicherheitsaspekte der Blockchain gespeichert werden.

2019

BlockME: Modellierung von Blockchain-spezifischen Charakteristika

Falazi et al.[↑] berücksichtigen erstmals die technischen Eigenschaften der Blockchain und deren Einfluss auf die Blockchain-basierte Prozessausführung [45]. Im Speziellen wird das Problem der probabilistischen Finalität diskutiert, wodurch Transaktionen erst mit mehr und mehr gefundenen Nachfolgeblöcken eine wachsende Wahrscheinlichkeit bezüglich der finalen Persistenz in der Blockchain erhalten. Der vorgestellte BLOCKME-Ansatz ermöglicht es dem Modellierer, mit einer BPMN-Erweiterung auf derartige Blockchain-spezifische Eigenheiten zu reagieren. Zum Beispiel kann durch einen sogenannten *EnsureTransactionStateTask* konfiguriert werden, nach wie vielen gefundenen nachfolgenden Blöcken die Transaktion als sicher gelten soll und da-

2 https://ethereum.org/de/nft, besucht am 19.01.2024

mit der Prozessfluss fortgesetzt wird. Neben speziellen BPMN-Tasks werden auch verschiedene Ereignisse eingeführt, sodass beispielsweise auf invalidierte oder verwaiste Transaktionen mit speziellen eskalierenden Ausführungspfaden reagiert werden kann. Letztendlich ist dieser Ansatz nur indirekt zur Blockchain-basierten Prozessausführung zu zählen, da der Fokus hier auf der Integration Blockchain-basierter Dienste in die zentral-organisierte Prozessausführung liegt.

BlockME2: Spezifikation von Smart Contract-Funktionalitäten im Prozessmodell

Eine Erweiterung dazu wird mit BlockME2 ebenfalls von **Falazi et al.**[↑] vorgestellt [46]. Darin wird auf die Einschränkungen von BlockME reagiert, insbesondere bezüglich Blockchains mit probabilistischer Finalität oder nicht-linearer Datenstruktur. Für die Unterstützung von BFT-Blockchains wird eine flexiblere Finalitätsberechnung vorgestellt. Außerdem kann mit BlockME2 die operationale Perspektive durch den Aufruf beliebiger Smart Contract-Funktionen implementiert werden, indem im BPMN-Modell die entsprechenden Funktionen in den Annotationen von Aktivitäten spezifiziert werden. Durch die flexiblere Finalitätsberechnung einerseits, aber auch durch die Definition einer abstrakten *BlockchainAccessLayer* und der Implementierung von *Adaptern* ist es möglich, verschiedene Blockchainsysteme wie Ethereum oder Hyperledger innerhalb eines Prozessmodells anzusprechen. Für die Implementierung und Ausführung in verfügbaren WFMSs werden die BPMN-Erweiterungen in Standard-BPMN übersetzt. Auch hier liegt der Fokus bei der Integration von externen Blockchain-Diensten, anstatt auf der zwischenbetrieblichen Prozessausführung nach Decentralized Control.

BPMN Choreografie-Erweiterung zur Ausführung

Ladleif et al.[↑] diskutieren 2019 die Verwendung von BPMN-Prozessdiagrammen und BPMN-Choreografien für den Einsatz in der Blockchain-basierten Prozessausführung [96]. Trotz der Vorteile von BPMN-Prozessdiagrammen wie die weite Verbreitung, argumentieren die Autoren für die Verwendung von Choreografien, zum Beispiel wegen des hohen Abstraktionsniveaus, welches bei zwischenbetrieblichen Kollaborationen gefordert ist [106, 187]. Allerdings fehlt bei BPMN-Choreografien eine detaillierte Ausführungssemantik, da vor der Entwicklung der Blockchain keine geeignete Infrastruktur für deren Ausführung verfügbar war und Choreografien stattdessen eher zur Koordination von ausführbaren privaten Prozessen eingesetzt wurden [187]. In ihrem Beitrag erweitern die Autoren die BPMN-Choreografien um notwendige Ausführungsdetails, zum Beispiel Datenobjekte, Events, Skript Tasks oder Sub-Choreografien, welche von BPMN-Prozessdiagrammen adaptiert sind. Die Autoren favorisieren also weiterhin das Paradigma des modellierten Nachrichtenaustauschprotokolls, obwohl mit der Blockchain-Technologie eine Infrastruktur integriert wird, mit der auch Prozessmodelle ausgeführt werden können.

Caterpillar

In [113] stellten **López-Pintado et al.**[↑] das Caterpillar-System als erstes Blockchain-basiertes WFMS im Detail vor, nachdem in [110] lediglich eine Demo der rudimentären Proof of Concept-Implemen-

tierung veröffentlicht ist. Caterpillar verwendet zur Modellierung BPMN-Prozessdiagramme und unterstützt dabei viele Notationselemente, darunter auch Events und Sub-Prozesse. Als Ausführungsplattform wird die Ethereum-Blockchain verwendet. Dadurch dass die Arbeit von Weber et al. [185] als Grundlage dient, verwendet auch Caterpillar einen kompilierenden Ansatz, das heißt, basierend auf einem Modell wird modellspezifischer Quelltext eines Smart Contracts generiert.

Organisationale Perspektive in Caterpillar

Eine Erweiterung von Caterpillar hinsichtlich der organisationalen Perspektive stellen **López-Pintado et al.**[↑] in [108] vor. Dadurch werden statt einmaliger Rollenzuweisungen zur Kompilierzeit auch die Zuordnungen von Rollen zu Aktivitäten zur Laufzeit unterstützt, um die Verantwortlichkeiten für die Ausführung dynamisch spezifizieren zu können. Für die Umsetzung wird eine auf Petri-Netzen basierende Sprache entwickelt, mit der die Zuweisungsrichtlinien spezifiziert werden können (*binding policy specification language*).

Interpretierender Ansatz in Caterpillar

Nach Vorbild des in Kapitel 5 vorgestellten und in [168, 170] veröffentlichten interpretierenden Ansatzes, ergänzen **López-Pintado et al.**[↑] in [109] das Caterpillar WFMS um einen BPMN-Interpreter. Dadurch sind zusätzlich dynamische Änderungen am Prozess zur Ausführungszeit möglich, um die Flexibilität des Systems zu erhöhen.

2020

Zeitperspektive in Caterpillar

Abid et al.[↑] erweitern Caterpillar um die Zeitperspektive durch die Einführung relativer Zeitbedingungen, welche zeitliche Bedingungen zwischen zwei Aktivitäten spezifizieren und absoluter Zeitbedingungen, welche die Start- und Endbedingungen einer einzelnen Aktivität einschränken [4]. Die Umsetzung erfolgt durch die Generierung zusätzlicher *Time-Guard*-Funktionen in den Smart Contract-Funktionen von Aktivitäten, welche mit Timer-Events annotiert sind.

Trennung von Zuständigkeiten beim LCW-Prinzip

Adams el al.[↑] diskutieren in [5] eine Trennung der Prozessausführungsfunktionalität von der umsetzenden Blockchain-Infrastruktur. Die verfolgten Entwurfsprinzipien, welche nachfolgend erklärt werden, umfassen unter anderem die Unterscheidung zwischen privaten und öffentlichen Prozessen, *Compartmentalisation of Interactions* (deutsch: Kompartimentierung von Interaktionen) und die Trennung von Zuständigkeiten. Die strikte Trennung des zwischenbetrieblichen öffentlichen Prozesses und der lokalen, privaten Prozesse bietet den Teilnehmern größere Flexibilität, die internen Prozesse bei Bedarf einfach zu ändern. Durch die Kompartimentierung von Interaktionen soll explizit verhindert werden, dass alle Teilnehmer der Kollaboration einen globalen Prozessstatus teilen. Somit wird ein hohes Maß an Diskretion und Vertraulichkeit von Informationen erreicht, jedoch auf Kosten einer hohen Transparenz. Um dem nachzukommen, existiert auch kein globales Prozessmodell, stattdessen werden

lediglich lokale Prozesse und deren Interaktionen verwendet, welche ähnlich einer Choreografie modelliert sind. Die Trennung der Zuständigkeiten hebt die feste Bindung an ein bestimmtes Blockchain-Protokoll oder an ein bestimmtes Prozessausführungssystem auf. Die Plattform-Heterogenität erlaubt darüber hinaus den gleichzeitigen Einsatz mehrerer Blockchain-Protokolle. Für die Modellierung und Ausführung wird YAWL verwendet und als Blockchain-System wird in der Demonstration Hyperledger Fabric angebunden. Zusammengefasst existiert bei diesem Ansatz kein gemeinsames, globales Prozessmodell und keine gemeinsame Blockchain-Infrastruktur und es handelt sich demnach eher um eine Implementierung des Loosely Coupled Workflow-Prinzips. Dies bedeutet, dass die Vorteile der traditionellen Prozessausführung oder von Decentralized Control wie ein globaler Prozessstatus oder kollaborationsweite Arbeitslisten hier nicht verfügbar sind.

Blockchain zur passiven Event Log-Dokumentation

Der Ansatz von **Alves et al.**[↑] [6] ist nicht als Ansatz der Blockchain-basierten Prozessausführung oder Decentralized Control im engeren Sinne zu klassifizieren, da die Ausführung in einem zentral verwalteten WFMS organisiert ist. Die Autoren beschreiben einen Anwendungsfall im Bereich der geophysischen Datenerfassung, an dem mehrere nicht vertrauenswürdige Teilnehmer involviert sind. Darin stellt ein geophysisches Institut das einzige involvierte Camunda-System zur Verfügung, über welches die Klienten alle Daten über ein Camunda-Formular eintragen können. Um die Verlässlichkeit, Integrität und Nachvollziehbarkeit der eingegebenen (Prozess)-Daten zu gewährleisten, werden alle Prozessaktionen mithilfe eines Camunda-Plug-Ins auf einer Hyperledger-Blockchain gespeichert. Die Blockchain wird in diesem Ansatz lediglich als passiver Datenspeicher genutzt und nicht zur aktiven Steuerung eines zwischenbetrieblichen Prozesses oder als Kommunikationsebene.

transaktionale Geschäftsprozesse in Caterpillar

Eine nächste Erweiterung von Caterpillar stellen **Brahem et al.**[↑] vor, welche das Konzept von *Transactional Business Processes* (deutsch: transaktionalen Geschäftsprozessen) integriert [16]. Bei transaktionalen Geschäftsprozessen wird unter anderem berücksichtigt, dass das Fehlschlagen von Aktivitäten Auswirkungen auf andere Aktivitäten haben kann. Zum Beispiel müssen bei einem Bestellprozess die vorangegangenen Aktivitäten zum Auswählen von Paketinhalten zurückgesetzt werden, wenn die Aktivität zum Bestätigen der Auswahl fehlschlägt, oder sollte die Aktivität zum Bezahlen mit Kreditkarte fehlschlagen, wird direkt Aktivität Barzahlung aktiviert. Auf Modellierungsebene wird der Transaktionsfluss im Prozessmodell ergänzt, auf konzeptioneller Ebene wird der Aktivitätslebenszyklus erweitert und für die Implementierung werden die BPMN-zu-Solidity-Transpiler erweitert, um die entsprechende Funktionalität im Quelltext der Smart Contracts zu generieren.

Lebenszyklus von BPMN Choreografien

Bezüglich der fehlenden Ausführungssemantik von BPMN-Choreografien [96] stellen **Corradini et al.**[↑] das ChorChain-System vor, welches sich auf die Unterstützung des kompletten Lebenszyklus einer Choreografie fokussiert [27]. Der Lebenszyklus wird durch die Phasen Modellierung, Veröffentlichung, Instanziierung, Bereitstellung und Ausführung definiert. Dafür wird unter anderem ein Repositorium für Choreografien eingeführt, welches eine wiederholte Instanziierung der Modelle erlaubt. Das ChorChain-System basiert auf dem Ethereum-Protokoll.

BFT-Algorithmen statt Proof of Work Blockchains

Evermann et al.[↑] studieren die Verwendung alternativer Konsensmechanismen für die Blockchain-basierte Prozessausführung [43, 44]. Insbesondere wird hervorgehoben, dass BFT-basierte Verfahren mit unmittelbarer und finaler Konsensfindung überzeugen. Dies wird jedoch auf Kosten einer geringeren Skalierbarkeit im Vergleich zu globalen Proof of Work-Blockchains mit probabilistischer Finalität erreicht. Die Anforderung an eine derart hohe Skalierbarkeit wird im Bereich der Prozessausführung allerdings infrage gestellt, weswegen BFT-Blockchains zu bevorzugen sind. Die Architektur zeichnet sich durch eine Trennung von Zuständigkeiten bezüglich der folgenden drei Dienste aus. Der *Ordering Service* synchronisiert konkurrierende Nachrichten auf Basis von BFT-SMaRt [12], der *Block Service* speichert und verifiziert die Blockchain und eine eigens entwickelte, prototypische *Workflow Engine* mit sehr eingeschränktem Funktionsumfang wird für die Prozessausführung eingesetzt.

Dezentralisierung durch Abstimmungsmechanismus

Der Ansatz von **Klinger et al.**[↑] verwendet BPMN-Kollaborationsdiagramme und integriert einen Abstimmungsmechanismus, durch welchen eine stärkere Dezentralisierung der Prozessausführung ermöglicht wird [84]. Denn im Vergleich zu anderen Ansätzen (zum Beispiel [170], welcher in Kapitel 5 vorgestellt wird) wird keine zentrale Entität für administrative Aufgaben, wie etwa der Bereitstellung, benötigt, was auch für eine fairere Aufteilung der Transaktionskosten sorgt. Die Unterscheidung zwischen einem *Collaboration Contract* und den *Participant Contracts* erleichtert weiterhin die Anbindung von Off-Chain-Komponenten, da für jeden Teilnehmer ein eigener Smart Contract existiert.

Abstimmungsbasierte Evolution von Prozessmodellen

In einer Erweiterung führen **Klinger et al.**[↑] ein Upgrade-Konzept ein, welches die Verwaltung von verschiedenen Versionen eines Prozesses erlaubt [85]. Für die Implementierung werden verschiedene Entwurfsmuster aus der Softwareentwicklung evaluiert.

Kompilierung von Choreografien für verschiedene Blockchains

Ladleif et al.[↑] schlagen einen Multi-Chain Ansatz vor, um divergierenden Anforderungen an Vertraulichkeit und Datensicherheit im Bereich der Blockchain-basierten Prozessausführung zu entgegnen [94]. Im Mantichor-System werden BPMN-Choreografien als Eingabedaten verwendet, welche durch Adapterimplementierungen für verschiedene Blockchains in den entsprechenden ausführbaren Smart Contract-Quelltext übersetzt werden können. Damit wird die Umset-

zung einer Kollaboration über mehrere Blockchains hinweg unterstützt.

konzeptionelle Diskussion der Zeitperspektive

In [95] beschäftigen sich die Autoren **Ladleif et al.**[↑] in einer rein konzeptionellen Arbeit mit der Zeitperspektive vor dem Hintergrund der *Closed World Assumption* (deutsch: Annahme zur Weltabgeschlossenheit), also dass keine globale eindeutige Uhr innerhalb der Blockchain, einem verteilten System, existiert. Im Gegensatz zum eher implementierungsfokussierten Beitrag von Abid et al. [4] werden verschiedene Möglichkeiten zur Ableitung von Zeit-Metriken während der Blockchain-basierten Prozessausführung diskutiert, zum Beispiel die Blockzeit, die Blocknummer oder Oracles. Diese werden anhand verschiedener Kriterien, darunter Genauigkeit, Kosten oder Sicherheit, bewertet und anschließend für den Einsatz in der Prozessausführung evaluiert.

Daten-zentrierter Ansatz

Lichtenstein et al.[↑] kümmern sich um die Wiederverwendung von Datenobjekten bei der Ausführung von Choreografien auf der Ethereum-Blockchain [104]. In diesem Daten-zentrischen Ansatz werden Annotation im Modell verwendet, um Datenobjekte zu definieren und die Nachbedingungen für die Ausführung eines Choreografie-Tasks zu spezifizieren. Für die Wiederverwendbarkeit wird ein *DataObjectStore* (deutsch: Datenobjektspeicher) eingeführt.

abstrakte Beschreibung eines Prozessausführungs-artefakts

Nagano et al.[↑] stellen in [131, 132] ein System zur Prozessausführung auf Basis von Hyperledger vor. Dabei werden nicht näher beschriebene *Workflow Definitions* in einem Smart Contract verwaltet, während ein zweiter Smart Contract für die Abarbeitung der Prozesse zuständig ist. Das System berücksichtigt keine Konzepte wie Flexibilität oder die organisationale Prozessperspektive.

eigene DLT-Implementierung

Basierend auf ihrer eigenen DLT-Implementierung, LabChain, integrieren **Osterland et al.**[↑] die Ausführungslogik der Prozessmodellierungssprache direkt in die Validationsroutine des DLT-Netzwerkknotens [137]. Die Transaktionen des Systems repräsentieren die auszuführenden Prozessaktionen, wobei die Validierung der Prozessaktion direkt über die Validierung der Transaktion stattfindet. Letzlich ist die Idee auf konzeptioneller Ebene ähnlich zu anderen Ansätzen, denn die Netzwerkknoten entscheiden über die Prozesskonformität, bevor eine Transaktion in die Blockchain-Datenstruktur integriert wird. Der Unterschied zu anderen Ansätzen ist, dass die Prozesskonformität nicht über einen modellspezifischen, generierten Smart Contract oder einen Interpreter-Smart Contract geprüft wird, sondern direkt im *Kernel*, also der Anwendung des Netzwerkprotokolls.

BPMN und Hyperledger: Ausführung, Analyse und Laufzeitänderungen

Die BPMChain [152] von **Schinle et al.**[↑] ist ein kompilierender Ansatz zur Ausführung eines BPMN-Prozessmodells auf der Hyperledger-Blockchain, welcher im Gegensatz zu der ebenfalls auf Hyperledger basierenden Arbeit von Nakamura et al. [134] auch Laufzeitänderungen am Prozess unterstützt. Des Weiteren unterstützt die

BPMChain auch weitere Phasen des Prozesslebenszyklus, etwa die Analyse.

2021

flexible Wahl der Blockchain in Multi-Chain

Corradini et al.[↑] beschäftigen sich ebenfalls mit den unterschiedlichen Anforderungen aus der Prozessdomäne an ausführende Blockchain-Infrastrukturen und evaluieren dabei die unterschiedlichen Konfigurationsmöglichkeiten bezüglich der Ethereum-Blockchain und der Hyperledger-Blockchain. Das vorgestellte MULTI-CHAIN-System [28] kann basierend auf einer BPMN-Choreografie entweder den Smart Contract-Quelltext für eine Ethereum-Blockchain oder den Chaincode für die Verwendung in einer Hyperledger-Blockchain produzieren und fällt damit in die Kategorie der kompilierenden Ansätze. Damit ist Multi-Chain in der Wahl der Blockchain-Systeme flexibel, was jedoch nur zur Kompilierzeit gilt und nicht zur Laufzeit. Während das Mantichor-System [94] eine Integration verschiedener Blockchains in die Ausführung einer Prozessinstanz fokussiert, wird dies hier nicht unterstützt. Multi-Chain erstellt dafür zusätzlich automatisch die von Hyperledger erforderlichen Konfigurationen.

flexible Interpretation in CoBuP

Loukil et al.[↑] beschäftigen sich mit dem Thema Flexibilität und stellen das CoBuP-System (Collaborative Business Process Execution Architecture) (deutsch: Architektur zur Ausführung kollaborativer Geschäftsprozesse) vor [111], welches mit einem BPMN-Interpreter dafür sorgt, dass wiederholte Smart Contract-Bereitstellungen bei Änderungen am Prozessmodell nicht mehr notwendig sind. Mit der Architektur wird ebenfalls eine Evaluation veröffentlicht, welche ausgewählte kompilierende [53, 185] und interpretierende [109, 170] Ansätze mit CuBuP bezüglich der GAS-Kosten anhand von zwei Beispielprozessen vergleicht. Entgegen zu dem in Kapitel 5 vorgestellten Artefakt, bei dem der Interpreter und die Datenstrukturen für die Prozessausführung in einem gemeinsamen Smart Contract bereitgestellt werden, separiert CuBuP Interpreter und Datenstrukturen. Die erreichte Flexibilität ist mit höheren Initialkosten verbunden, allerdings reduzieren sich die Anzahl der Transaktionen. Der Evaluation nach amortisieren sich die Kosten nach 91 Instanziierungen eines Beispielmodells, ab welchen CuBuP kostengünstiger arbeitet.

austauschbare Tokens in Lorikeet

In [112] erweitern **Lu et al.**[↑] das LORIKEET-System, eine integrierte Lösung von Prozessausführung und Asset-Management. Das Asset-Management kann als Implementierung der Datenperspektive im Blockchain-Kontext betrachtet werden, denn Assets oder Tokens sind auf der Blockchain über Smart Contracts verwaltete Datenobjekte mit vorgegebener Struktur. Die Integration in Lorikeet verfolgt einen modellgetriebenen Ansatz mit einer BPMN-Erweiterung, bei dem das Prozessmodell, die Assets sowie deren Interaktionen modelliert werden, um anschließend daraus die entsprechenden Smart Contracts auto-

matisch zu generieren. Die Assets unterscheiden sich in austauschbare Token wie Kryptowährungen oder nicht austauschbare Token wie Diamanten oder Grundstücke, welche durch ihre physischen Eigenschaften oder geografische Lage eindeutig bestimmt sind. Das Beispiel von Getreideverkäufen in [176] wird hier um austauschbare Tokens ergänzt, sodass der Prozess der Eingangskontrolle (Wiegen und Qualitätskontrolle), der Erstellung eines nicht austauschbaren Tokens für das Getreide sowie deren Übertragung bei Bezahlung von austauschbaren Tokens, vollständig auf der Blockchain automatisiert werden kann.

Ganzheitlicher Ansatz basierend auf Sturm et al.

Einen ganzheitlichen Ansatz bezüglich dem Ausführen, Überwachen und der Analyse von zwischenbetrieblichen Prozessen stellen **Morales-Sandoval et al.**[↑] vor [114]. Dabei erweitern Sie den Ansatz des in Kapitel 5 vorgestellten Artefakts [168, 170] wie folgt. In dem bereitgestellten Smart Contract können auch mehrere Prozessinstanzen verwaltet werden. Das heißt, dass nicht nur zwischen Smart Contract und Prozessmodell eine 1:M-Beziehung besteht, sondern auch zwischen Smart Contract und Prozessinstanz. Bei Sturm et al. [168] liegen die Kardinalitäten hingegen bei 1:M beziehungsweise 1:1 und bei kompilierenden Ansätzen wie etwa bei Weber et al. [185] lediglich bei 1:1 und 1:1. Grundlegende Funktionen wie `addCollaborator`, `setTaskOnCompleted` sowie das Prinzip der *Requirements* bleiben jedoch erhalten. Im Gegensatz zu anderen in der Arbeit evaluierten Ansätzen [86, 128, 168, 185], integrieren die Autoren sowohl die Ausführung als auch die Analyse.

dynamische Zusammensetzung von Fragmenten

Im Ansatz von **Zhang et al.**[↑] werden Prozessmodellfragmente definiert, welche zur Laufzeit ausgewählt und komplett dynamisch zu einem zwischenbetrieblichen Prozess zusammengesetzt werden [197]. Wie im Anwendungsfall beschrieben, ist diese Laufzeitevolution von Prozessen essenziell bei der Umsetzung von Lieferketten, da während der Modellierung noch Unsicherheiten herrschen, welche erst zur Laufzeit aufgelöst werden können. Im Prozessbeispiel der Milchproduktion kann dieser zu einem Joghurt-Produktionsprozess evolviert werden, etwa bei veränderter Nachfrage während der Prozessausführung oder unvorhergesehener Überproduktion des Rohstoffs Milch.

2022

Vertrauen in ChorChain durch Auditing

In [29] erweitern **Corradini et al.**[↑] ihr ChorChain-System [27] um eine Auditingkomponente, um gespeicherte Prozessdaten für ein besseres Vertrauen in das System auf der Blockchain besser nachvollziehen zu können. Außerdem erweitern sie die Ausdrucksmächtigkeit von *messages* und *guards*, um unter anderem komplexere Kondition an ausgehenden Pfaden von exklusiven Gattern zu spezifizieren.

Flexibilität in Caterpillar

Aufbauend auf ihrer Arbeit in [108] erweitern **López-Pintado et al.**[↑] in [107] CATERPILLAR um folgende Funktionen. Um die Prozessflexibilität zu steigern, wird die Auswahl von Sub-Prozessen zur Laufzeit unterstützt, zudem können auszuführende Prozesspfade basierend auf einem Abstimmungsmechanismus dynamisch bestimmt werden.

Zeit und inklusives OR in Caterpillar

Naha et al.[↑] erweitern ebenfalls das CATERPILLAR-System um die Unterstützung sogenannter Timer-Events und dem inklusiven Oder-Gatter in der funktionalen Prozessperspektive [175]. Dabei werden unter anderem Laufzeitänderungen bezüglich der Timer-Konfiguration mit rollenbasierter Zugriffskontrolle unterstützt. Im Hinblick auf die Implementierung von inklusiven Gattern wird ein vorangegangener User Task angenommen, bei dem die Entscheidung über die auszuführenden Pfade getroffen wird. Über die Anzahl der Pfade kann beim entsprechenden zusammenführenden Gatter auf alle Tokens gewartet werden. Bei der ähnlichen Arbeit von Abid et al. [4], worin ebenfalls Caterpillar um die Zeitperspektive erweitert wird, identifizierten Naha et al. einen fehlenden Bezug von implementierten Zeitbedingungen und unterschiedlichen Timer-Events, welche im BPMN-Standard definiert sind.

2023

WFMS für Validierung von Transaktionen in Hyperledger

Wang et al.[↑] modifizieren das WFMS Activiti, um es direkt in das Hyperledger Projekt einzubinden und so für die Validierung von Transaktionen einzusetzen [183]. Statt die Prozesslogik in einem Smart Contract zu spezifizieren, wird also das WFMS zur Prüfung der Konformität eingesetzt. Dafür wird die Ausführung in drei Phasen untergliedert. In der *Endorsement*-Phase (deutsch: Billigung) wird die Konformität der von Transaktionen beschriebenen Prozessausführungsschritte über eine Simulation geprüft. In der *Consensus*-Phase (deutsch: Einigung) wird die Anzahl der Bestätigungen geprüft und bezüglich einer Endorsement-Strategie die entsprechende Transaktion gegebenenfalls auf die Blockchain geschrieben. In der *Committing*-Phase (deutsch: verbindliche Festlegung) führen die Activiti-Systeme schließlich die Aktion aus und aktualisieren den lokalen Prozessstatus. Besonderer Fokus in dem Ansatz ist die Integration von externen Diensten. Danach müssen Teilnehmer angebotene Dienste zuvor über die Blockchain registrieren, sodass eine Vorabprüfung stattfinden kann, bevor diese in der Prozessausführung eingesetzt werden.

DCR-Graphen auf Algorand-Blockchain

Xu et al.[↑] stellen in [192] einen Ansatz zur Ausführung von DCR-Graphen auf der Algorand-Blockchain vor. Damit erreichen sie eine kostengünstigere Ausführung mit schnellerer Finalität. Eine Herausforderung bei der Entwicklung stellt die eingeschränkte Ausdrucksmächtigkeit der in der TEAL-Sprache verfassten Smart Contracts dar.

8.3 EINORDNUNG DER ARTEFAKTE IN DEN FORSCHUNGSBEREICH

In diesem Kapitel erfolgt eine Abgrenzung der in dieser Arbeit vorgestellten Artefakte zur Blockchain-basierten Prozessausführung von den bereits existierenden Ansätzen. Dazu werden die als sehr relevant eingestuften und in Tabelle 7 grün markierten Publikationen in Tabelle 8 kategorisiert.[3] Kategorien mit niedriger Bezifferung umfassen eher rudimentäre Umsetzungen in diesem Forschungsbereich, welche sich aus BPM-Sicht beispielsweise unflexibler Techniken der Quelltextgenerierung bedienen. Die höheren Kategorien berücksichtigen verstärkt die Prinzipien der Prozessausführung und integrieren zudem verbesserte Konzepte, etwa zur Steigerung der Flexibilität.

Kategorie 0: Rein konzeptionelle Arbeiten

KATEGORISIERUNG DER ARBEITEN In Kategorie 0 findet sich lediglich die Arbeit von Ladleif et al. [95], worin die Integration der Zeitperspektive in die Blockchain-basierte Prozessausführung konzeptionell aufgearbeitet wird. Die Einordnung erfolgt aufgrund eines fehlenden konzeptionellen Systementwurfs, einer fehlenden Implementierung und der eingeschränkten Fokussierung auf die Zeitperspektive.

Kategorie 1: Blockchain nicht als Koordinator

Kategorie 1 umfasst die Ansätze, welche die Blockchain nicht als globales Koordinationsinstrument einsetzen. Dazu zählen die Arbeiten von Falazi et al. [45, 46], welche im BlockME-Ansatz über eine traditionelle Prozessausführung die Aufrufe von Smart Contract-Funktionen koordinieren. Bei Alves et al. [6] steuert ein zentralisiertes Camunda WFMS die Prozessausführung, während die Blockchain alle Ereignisse lediglich protokolliert. Adams et al. schließen in [5] einen globalen Prozessstatus auf der Blockchain ebenfalls per Anforderung explizit aus und forcieren eher eine Automatisierung des Loosely Coupled Workflow-Prinzips, wonach die Benachrichtigung des nächsten Prozessteilnehmers über die Blockchain abgewickelt wird. Lichtenstein et al. verwalten in [104] den Prozessstatus ebenfalls lediglich implizit auf der Blockchain über Datenobjekte, dessen Status über die Umsetzung von BPMN-Choreografien außerhalb der Blockchain aktualisiert wird. Dadurch dass die Blockchain nicht als zentraler Koordinator fungiert, ist es in den Ansätzen von Falazi et al. und Adams et al. möglich, dass verschiedene Teile der zwischenbetrieblichen Kollaboration über unterschiedliche Blockchains abgewickelt werden. Dies ist im Konzept von dpex nicht möglich, allerdings wird das bei einer Konzeptionierung der Blockchain als zentrale Kommunikationsinfrastruktur auch nicht als Anforderung definiert.

Kategorie 2: kompilierende, nachrichtenbasierte Ansätze

Kategorie 2 sortiert die Ansätze, welche einerseits eine modellgetriebene, Quelltext-generierende Strategie verfolgen statt einen Prozess mittels eines Interpreters auszuführen. Andererseits verwenden

3 ausgenommen den Arbeiten von Nagano et al. [131, 132] aufgrund unzureichender Beschreibungen

die Ansätze der Kategorie 2 ein nachrichtenbasiertes Modell der Kollaboration, zum Beispiel eine BPMN-Choreografie, wodurch eine Prozessausführung im Sinne von Kapitel 2.3 schwer umzusetzen ist. Einzuordnen sind hier die ersten Vorschläge von Weber et al. [185], Ladleif et al. [96] oder ChorChain von Corradini et al. [27, 29]. Zum Teil werden Erweiterungen zur Generierung von Quelltext für unterschiedliche Blockchains vorgestellt, zum Beispiel Manti-Chor [94] von Ladleif et al. oder Multi-Chain [28] von Corradini et al. Dies erhöht die Flexibilität ähnlich zum dpex-Framework, indem die Ansätze für verschiedenen Zielsysteme verwendet werden können. Während dpex allerdings extra Schnittstellen dafür bereitstellt, ist eine Unterstützung weiterer Protokolle hier mit großem Implementierungsaufwand verbunden. Bei den Ansätzen von Klinger et al. [84, 85] erfolgt die Quelltextgenerierung basierend auf BPMN-Kollaborationsdiagrammen.

Kategorie 3: kompilierende, prozessbasierte Ansätze

Kategorie 3 gruppiert Veröffentlichungen, welche ebenfalls mit der Generierung von Quelltext arbeiten, jedoch prozessbasierte Diagramme als Eingabedaten verwenden und in diesem Sinne dem dpex-System und den Prozessausführungsprinzipien einen Schritt ähnlicher sind. Zu diesen Ansätzen zählen insbesondere Caterpillar [110] sowie die zugrundeliegende Arbeit von García-Bañuelos et al. [53]. Außerdem zählen dazu die Erweiterungen für Caterpillar von Mercenne et al. [122], López-Pintado et al. [108, 113], Abid et al. [4], Brahem et al. [16] und Naha et al. [175]. Diese integrieren die organisationale und zeitliche Perspektive oder transaktionale Belange, ohne dabei die grundlegende Funktionsweise von Caterpillar weiterzuentwickeln. Weiterhin zu nennen ist das Lorikeet-System, von Tran et al. [176] beziehungsweise Lu et al. [112], worin unter anderem durch eine Erweiterung von BPMN auch Quelltext für das Asset-Management innerhalb der Prozessausführung generiert werden kann. Weniger häufig eingesetzte Technologien verbindet der Ansatz von Xu et al. [193], welcher Quelltext zur Umsetzung von DCR-Graphen auf der Algorand-Blockchain generiert. Die Nachteile dieser kompilierenden Ansätze sind ausführlich in Kapitel 5.6 beziehungsweise Kapitel 6.3 aufgearbeitet.

Kategorie 4: interpretierende Ansätze

In Kategorie 4 werden die Konzepte nach dem Vorbild des interpretierenden Ansatzes eingruppiert. Dazu zählen allen voran die in Kapitel 5 vorgestellten Arbeiten von Sturm et al. [168–170] und darauf aufbauende Arbeiten von Morales-Sandoval et al. [114], mit einer optimierten Ressourcenverwaltung und der Bereitstellung von Ereignisprotokollen sowie von López-Pintado et al. [109], welche das Caterpillar-System um einen BPMN-Interpreter ergänzt. Eine weitere Caterpillar-Erweiterung von López-Pintado et al. ermöglicht zudem eine dynamische Laufzeitkonfiguration der auszuführenden Prozesse [107], ähnlich zum Ansatz von Zhang et al. [197], worin ebenfalls Prozessmodellfragmente dynamisch zur Laufzeit zu einem zwischenbetrieblichen Prozess zusammengesetzt werden. Die weiteren Arbeiten in Kategorie 4 diskutieren eine Ausführung von BPMN-Prozess-

diagrammen auf der Hyperledger-Blockchain (Nakamura et al. [134] und Schinle et al. [152]), die interpretierte Umsetzung von BPMN-Choreografien auf der Ethereum-Blockchain (Loukil et al. [111]) oder die Ausführung von DCR-Graphen, ebenfalls auf der Ethereum Blockchain (Madsen et al. [115]). Obwohl diese Ansätze eine interpretierte Ausführung von Prozessmodellen erlauben, sind die Prozessperspektiven beziehungsweise die wichtigsten Funktionalitäten von WFMSs bei diesen prototypischen Implementierungen nicht oder nur ungenügend umgesetzt.

Kategorie 5: fortgeschrittene Ansätze

Die Ansätze in Kategorie 5 verfolgen fortgeschrittene Ansätze, anstatt lediglich eine Form von Interpreter-Smart Contract auf einer Blockchain bereitzustellen, um darüber die Blockchain-basierte Prozessausführung umzusetzen. Evermann et al. diskutieren zur Synchronisierung verschiedener Teilnehmer den Einsatz von BFT-Algorithmen anstelle von Blockchains, welche Proof of Work als Konsensmechanismus verwenden. Da BFT-Algorithmen die Prozesskonformität nicht überprüfen können, integrieren die Autoren für die Interpretation der Prozesse eine externe BPM-Komponente wie YAWL [43] oder eine eigene Implementierung [44]. Auch im dpex-Framework erfolgt in ähnlicher Weise die Interpretation von Prozessen unabhängig von der Synchronisierung der Nachrichten. Zusätzlich erlaubt dpex allerdings variable Konfigurationsmöglichkeiten für verschiedene Kollaborationen und eine einfache Erweiterung und Anbindung bestehender BPM- und SCI-Systeme. Osterland et al. stellen als Alternative zur Ethereum-Blockchain eine eigene Distributed Ledger-Implementierung vor [137]. Dabei integrieren sie die Überprüfung auf Prozesskonformität in die Validationsroutine des Knotens. Konzeptionell unterscheidet sich der Ablauf nicht stark von der Ethereum-basierten Ausführung mit Smart Contracts, denn in beiden Fällen werden eingehende Nachrichten als Transaktionen hinsichtlich ihrer Prozesskonformität überprüft. Allerdings ist durch die enge Verknüpfung zwischen Ausführungssemantik und Validierungsroutine keine klare Trennung möglich, wodurch die Flexibilität bei diesem Ansatz stark eingeschränkt ist. Wang et al. verfolgen in [183] einen ähnlichen Ansatz, ohne jedoch eine manuelle Implementierung einer Blockchain aufzubauen. Stattdessen integrieren die Autoren ein existierendes WFMS direkt in die Validationsroutine eines Hyperledger-Knotens. Dadurch erfolgt zwar einerseits eine Schichtentrennung von BPM und SCI, allerdings bietet sich bei diesem gesamtheitlichen System keine Flexibilität hinsichtlich der Freiheit in der Auswahl oder Austauschbarkeit der beteiligten Systeme.

Kategorie 6: dpex

Das in Kategorie 6 einzuordnende dpex-Framework treibt eine vollkommene Entkopplung der BPM- und SCI-Domäne voran. Dies erlaubt eine einfache Integration von WFMSs mit breitem Funktionsumfang in die zwischenbetriebliche Prozessausführung. Außerdem können verschiedenste Blockchain-basierte oder traditionelle BFT-Algo-

rithmen für die Koordination der Systeme angebunden werden. Zuletzt ist es möglich, die angebundenen Systeme in den Adapterimplementierungen für bestimmte Anwendungsfälle anzupassen und zu erweitern, beispielsweise in Hinblick auf die Unterstützung der Prozessperspektiven. Durch die Konzeption als Middleware und Plattform ist die Anbindung und Erweiterung von Systemen nicht mit großem Implementierungsaufwand verbunden. Dies wird auch durch die Wiederverwendbarkeit der Adapter ermöglicht.

Beeinflussende Arbeiten von dpex

GEWONNENE ERKENNTNISSE Zusammengefasst beeinflussen also folgende Arbeiten das dpex-Framework. Der BlockME-Ansatz [45, 46] ermöglicht die Anbindung unterschiedlicher Blockchains durch die Definition eines *Abstract Blockchain Layer*s, verwendet die Blockchain allerdings nicht als globalen Koordinator. Alves et al. [6] integriert die Blockchain und ein externes WFMS, verwendet aber das WFMS statt der Blockchain zur Steuerung des Prozesses. Adams et al. [5] schlägt eine BPM-SCI-Schichtentrennung vor, das Konzept schließt aber einen globalen Prozessstatus aus. Evermann et al. [43, 44] unterscheidet auch zwischen Prozessausführung eines globalen Prozesses und Nachrichtensynchronisierung, allerdings ist das Konzept unflexibel in der Auswahl der Systeme, ähnlich zum Ansatz von Osterland et al. [137].

Nicht unterstützte Funktionalitäten in dpex

Eine Reihe von Ansätzen berücksichtigen allerdings noch fortgeschrittene Konzepte und Ideen, welche im dpex-Framework aktuell noch nicht umgesetzt sind. Bei dpex können die Prozessmodelle zur Ausführungszeit nicht verändert werden und derartige Flexibilität kann lediglich zum Beispiel durch XOR-Gatter umgesetzt werden. Die Zeitperspektive kann bei dpex zumindest aus lokaler Sicht implementiert werden, allerdings erfolgt keine globale Überprüfung der Zeitbedingungen wie von Ladleif et al. [95] diskutiert und von Naha et al. [175] in Caterpillar implementiert. Auch Blockchain-spezifische Funktionalitäten wie das Asset-Management im Lorikeet-System [112, 176] oder das treuhänderische Verwalten von Kryptowährungen wie von Weber et al. [185] vorgeschlagen, ist durch die Entkopplung von der Ethereum-Blockchain in dpex nicht direkt möglich und muss manuell im Prozess implementiert werden. Weitere Funktionalitäten, welche aktuell nicht in dpex umgesetzt sind, umfassen die Versionierung [85] oder Evolution [111] von Prozessmodellen, die simultane Koordination über mehrere Blockchain-Protokolle [46], automatische Konfigurationen der Blockchains [28] oder die Generierung von grafischen Benutzerschnittstellen [134].

Abschließend fasst Tabelle 8 die Arbeiten in den hier diskutierten Kategorien mit dem jeweiligen Kernbetrag noch einmal zusammen.

Jahr	P	Veröffentlichung	System	Ansatz	Modell	SCI	Beitrag
2020	◐	Ladleif et al. [95]			KBR	ETH	Evaluation möglicher Zeit-Metriken von Blockchains und deren Einsatz für die Prozessausführung; konzeptionelle Arbeit
2019	○	Falazi et al. [45]	BlockME		BPMN++	ETH & HL	Integration Blockchain-spezifischer Charakteristika (Finalität) in das Prozessmodell über BPMN Erweiterung; Abstraktion von Blockchain-Schicht
2019	○	Falazi et al. [46]	BlockME2		BPMN++	ETH & HL	Spezifikation von Smart Contract-Funktionalitäten im Prozessmodell über BPMN Erweiterung (neuer Tasktyp); Anbindung von Proof of Authority-Blockchains über generische Finality-Metrik
2020	○	Alves et al. [6]			BPMN	HL	Plug-In für Camunda, um Ausführungshistorie auf Hyperledger-Blockchain zu schreiben
2020	◐	Adams et al. [5]			YAWL	ETH & HL	Schichtentrennung bezüglich Blockchain und Prozessausführung, aber Vertraulichkeit statt globaler Prozessstatus
2020	☍	Lichtenstein et al. [104]			CHOR	ETH	Daten-zentrischer Ansatz für Choreografien; Wiederverwendung von Datenobjekte; keine explizite Ausführung
2016	○	Weber et al. [185]		kmp	CHOR	ETH	Umsetzung von BPMN Choreografien auf Ethereum
2019	◐	Ladleif et al. [96]		kmp	CHOR	ETH	BPMN Choreografie-Erweiterung um notwendige Detailinformationen zur Ausführung
2020	◐	Ladleif et al. [94]	Manti-Chor	kmp	CHOR	ETH | HL	Übersetzung in mehrere Protokolle aufgrund verschiedener Anforderungen bezüglich Vertraulichkeit und Sicherheit
2020	●	Corradini et al. [27]	ChorChain	kmp	CHOR	HL	Umsetzung des Choreografie-Lebenszyklus inklusive Choreografie-Repository für wiederholte Instanziierung
2022	●	Corradini et al. [29]	ChorChain	kmp	CHOR	ETH	Erweiterung von ChorChain [27] um Auditing-Komponente und der Ausdrucksmächtigkeit für `guards` und `messages`
2021	◐	Corradini et al. [28]	Multi-Chain	kmp	CHOR	ETH | HL	Übersetzung von Choreografien für Ethereum oder Hyperledger inklusive Konfiguration der Hyperledger-Blockchain
2020	◐	Klinger et al. [84]		kmp	KBR	ETH	Dezentralisierung der Bereitstellung durch Abstimmungsmechanismus für faire Kostenaufteilung (im Gegensatz zu [170]); 1 Smart Contract pro BPMN Pool für Off-Chain Integration
2020	●	Klinger et al. [85]		kmp	KBR	ETH	Abstimmungsmechanismus für Prozessversionierung
2017	○	Garcia-Banuelos et al. [53]		kmp	BPMN	ETH	Ausführung von BPMN Prozessdiagrammen auf Ethereum
2017		López-Pintado et al. [110]	Caterpillar	kmp	BPMN	ETH	Demo eines Blockchain-basierten WFMS (Caterpillar)
2019	◐	López-Pintado et al. [113]	Caterpillar	kmp	BPMN	ETH	Detailierte Beschreibung von Caterpillar inklusive Unterstützung vieler BPMN-Elemente
2019	○	López-Pintado et al. [108]	Caterpillar	kmp	BPMN++	ETH	Organisationale Perspektive in Caterpillar (wie [122]), inklusive dynamischer Rollenzuweisung zur Laufzeit
2018	○	Mercenne et al. [122]	Caterpillar	kmp	BPMN	ETH	Rollenkonzept-Erweiterung für Caterpillar
2020	◐	Abid et al. [4]	Caterpillar	kmp	BPMN	ETH	Zeit-Perspektive in Caterpillar, relative (Dauer zwischen Tasks) und absolute (Startzeitpunkt) Einschränkungen
2020	☍	Brahem et al. [16]	Caterpillar	kmp	BPMN++	ETH	Integration transaktionaler Geschäftsprozesse und erweiterter Aktivitätslebenszyklus in Caterpillar
2022	◐	Naha et al. [175]	Caterpillar	kmp	BPMN	ETH	Implementierung der Zeitperspektive und inklusiven Oder-Gatter in Caterpillar
2018	○	Tran et al. [176]	Lorikeet	kmp	BPMN++	ETH	Integration von Prozessausführung und Asset-Management in der Datenperspektive
2021	☍	Lu et al. [112]	Lorikeet	kmp	BPMN++	ETH	Erweiterung des integrierten Asset-Managements: BPMN-Erweiterung und Unterstützung von NFTs und $\mathcal{N}$FTs
2023	●	Xu et al. [192]		kmp	DCR	ALGO	Prozessausführung auf der Algorand-Blockchain mit limitierter Ausdrucksmächtigkeit der TEAL-Smart Contracts
2019		Sturm et al. [170]		ipr	BPMN	ETH	interpretierender Ansatz zur Prozessausführung auf Ethereum
2019		Sturm et al. [168]		ipr	BPMN	ETH	Erweiterung des interpretierenden Ansatzes [170] um organisationale Perspektive (3 WRP)
2020		Sturm et al. [169]		ipr	BPMN	ETH	Einführung Decentralized Control; Integration der Datenperspektive in [168, 170]
2021	●	Morales-Sandoval et al. [114]		ipr	BPMN	ETH	Optimierte Ausführung von [168] durch m:1:m-Kardinalität; Bereitstellung von XES-Logs während der Ausführung
2019	◐	López-Pintado et al. [109]	Caterpillar	ipr	BPMN	ETH	Interpretierender Ansatz in Caterpillar mit BPMN Interpreter für flexible Prozessausführung
2022	●	López-Pintado et al. [107]	Caterpillar	ipr	BPMN	ETH	Prozessflexibilität durch Auswahl von Sub-Prozessen und Ausführungspfaden; Konsens-basierte Kontrollflussflexibilität und Rollenbindung
2021	●	Zhang et al. [197]		ipr	BPMN	ETH	Definition von Prozessmodell-Fragmenten, welche zur Laufzeit dynamisch zusammengesetzt werden
2018	○	Nakamura et al. [134]		ipr	BPMN	HL	Interpretation von Prozessen auf Hyperledger über Zustandsautomaten
2020	◐	Schinle et al. [152]	BPMChain	ipr	BPMN	HL	Bereitstellung von Analysefunktionen durch direkt auslesbare Historie
2021	◐	Loukil et al. [111]	CuBuP	ipr	CHOR	ETH	Optimierte Ausführung durch Trennung von Ausführung und Datenstrukturen; Prozessevolution ohne erneute Smart Contract-Bereitstellung
2018	○	Madsen et al. [115]		ipr	DCR	ETH	Umsetzung von DCR-Graphen auf Ethereum
2020	●	Evermann et al. [43]		WFMS	YAWL	BFT	YAWL für Prozesskontrolldaten, Block Service und Ordering Service (BFT), keine kollaborationsweite Arbeitslisten
2020	◐	Evermann et al. [44]		WFMS	PN	BFT	BPM-BC-Schichtentrennung; Vorteile von BFT-Blockchains gegenüber PoW-Blockchains für die Prozessausführung
2020	☍	Osterland et al. [137]	LabChain			LabChain	Integration der Ausführungslogik in Validationsroutine des DLT-Knotens
2023	●	Wang et al. [183]			BPMN	HL	Modifikation des Activiti-WFMS zur Transaktionsvalidierung in Simulations-, Consensus- und Committing-Phase
2023		Sturm et al. [167]	dpex	∞	∞	∞	BPM-BC-Schichtentrennung; Anbindung externer Systeme über Adapter; BC nur für Kommunikation

Tabelle 8: Verwandte Arbeiten im Bereich der Blockchain-basierten Prozessausführung

8.4 WEITERE ANSÄTZE MIT INTERDISZIPLINÄRER ÜBERLAPPUNG

In diesem Kapitel werden weitere Arbeiten vorgestellt, welche die Integration der Blockchain-Technologie in das Prozessmanagement diskutieren. Während die Ansätze in Kapitel 8.2 vorrangig die Lebenszyklusphase der Prozessausführung fokussieren, kategorisieren sich die folgenden Arbeiten aus BPM-Sicht in allgemeine Challenge Paper, Prozessmodellierung, -analyse und -überwachung sowie die Blockchain-basierte Entscheidungsfindung auf der Blockchain, die Analyse von Sicherheitsaspekten und die Evaluation spezifischer Anwendungsfälle.

Challenge Paper

7 Forschungsrichtungen

Mendling et al.[↑] liefern mit [120] eine wegweisende Diskussion über die Verknüpfung von Prozessmanagement und der Blockchain-Technologie, ohne jedoch konkrete Lösungsvorschläge oder Artefakte vorzuschlagen. Darin identifizieren die 32 Autoren die Herausforderungen, welche die Anwendung der Blockchain-Technologie im BPM-Bereich mit sich bringt, unter anderem in Bezug auf die Phasen des Prozesslebenszyklus. Als Fazit werden sieben zu erwartende Forschungsrichtungen definiert, darunter Ausführungs- und Überwachungssysteme, neue Methoden zur Analyse und Spezifikation von Prozessen oder Herausforderungen bezüglich Evolution und Adaptation der Prozesse. Ähnliche Herausforderungen werden auch in [119] beschrieben.

Modellierung

Artefakt-zentrierte Paradigmen

In einer frühen Phase des interdisziplinären Forschungszweigs haben **Hull et al.**[↑] in [73] vorgeschlagen, eine Artefakt-zentrierte Sprache zur Modellierung von Kollaborationen zu entwerfen. Der Fokus der Arbeit ist dabei nicht die Sprache selbst, sondern eine Diskussion konzeptioneller Paradigmen. Da Distributed Ledger-Ansätze Daten und Prozesse gleichermaßen behandeln, würden sich Artefakt-zentrische Ansätze besser mit den Voraussetzungen in der Blockchain-Welt in Einklang bringen lassen, woraus sich Vorteile gegenüber den Prozess-zentrierten BPMN-Diagrammen ableiten lassen. Des Weiteren werden Vorteile bezüglich formaler Verifikationswerkzeuge diskutiert sowie weitere Herausforderungen angesprochen, zum Beispiel eine benötigte Abstraktion von Blockchain-spezifischen Implementierungsdetails oder die Frage nach der Favorisierung des kompilierenden Ansatzes [185] oder interpretierenden Ansatzes [170].

Artefakt-zentrierte Sprache für zwischenbetriebliche Prozesse

Amaral de Sousa et al.[↑] stellen mit B-MERODE eine dedizierte Modellierungssprache für zwischenbetriebliche Prozesse vor [160]. B-MERODE ist Artefakt-zentriert, das bedeutet, im Gegensatz zu Kon-

trollfluss-zentrierten Modellierungssprachen in BPMN rücken Artefakte oder Datenobjekte in den Vordergrund, während Operationen und Lebenszyklusphasen der Objekte darauf aufbauend definiert werden. Die Anforderungen an B-MERODE gruppieren sich in die *Business Object*-Dimension (Datenobjekte inklusive deren Kardinalitäten, Pflichtfelder und Zugriffskontrolle), die Lebenszyklus-Dimension (zeitliches Verhalten von Objekten basierend auf Zustandsübergangsdiagrammen und Wechselwirkungen zwischen Objekten), die Dimension der modellgetriebenen Softwareentwicklung (automatische Plattform-unabhängige Quelltextgenerierung) und die zwischenbetriebliche Dimension (Organisationale Entitäten, Rollen oder Entscheidungen). Die Modellierung erfolgt über die Erstellung des *Existent-Dependency*-Graphen, der alle Teilnehmer, Datenobjekte und deren Beziehungen beschreibt. Die *Object Event Table* beschreibt alle Ereignisse, worauf basierend ein Zustandsübergangsdiagramm erstellt wird. Danach können mit *Object Constraint Langauge*-Regeln zusätzliche Bedingungen definiert werden, die bei einer durch ein Ereignis ausgelösten Zustandsänderung von einem Objekt gelten müssen. Zuletzt spezifiziert die *Event-Participant Table* für jeden Objekttyp und jeden Teilnehmertyp oder Rolle, welche Ereignisse ausgelöst werden dürfen. Obwohl die Generierung von Smart Contract-Quelltext als Anforderung definiert wird, fokussiert sich die Arbeit auf die Vorstellung der Modellierungssprache. Die Evaluation erfolgt unter anderem durch einen Vergleich mit den ähnlichen Daten-getriebenen Ansätzen von Lu et al. [112, 176] (Lorikeet) und Lichtenstein et al. [104].

BPMN-Erweiterung bezüglich Sicherheit

Köpke et al.[↑] stellen 2023 mit SecBPMN2BC eine Methodik zur Modellierung von Prozessen vor, die sicherheitsrelevante Aspekte integriert [92]. Die Methodik baut auf BPMN-Prozessdiagrammen auf und erweitert den Standard um neue Modellierungselemente bezüglich der Blockchain-basierten Ausführung. Allerdings liefern die Autoren keine Implementierung mit und lassen die Umsetzung der Implementierungsphase offen für künftige Arbeiten.

Analyse

Logging in Smart Contracts

Der Ansatz von **Klinkmüller et al.**[↑] ermöglicht es, Informationen von beliebigen Smart Contracts in eine standardisierte Form zu extrahieren. Dafür werden in einem Manifest bestimmte Konfigurationsparameter festgelegt, welches in einer Extraktionskomponente für die Erstellung der benötigten Daten verwendet wird [86]. Zusätzlich produziert ein Generatormodul Solidity-Quelltext, welcher die Funktionalität zum Logging gemäß dem Manifest bereitstellt. Der Ansatz unterscheidet sich zu Mühlberger et al. [128] dadurch, dass hier kein Blockchain-basiertes Ausführungssystem für die Erstellung der benötigten Daten benutzt wird.

Extraktion der Ausführungshistorie

Mühlberger et al.[↑] befassen sich mit der Herausforderung, die relevanten Informationen für Analyseaufgaben, welche während der Prozessausführung auf Blockchain-Systemen generiert und als Transaktionen auf der Blockchain gespeichert werden, aufzubereiten und in eine standardisierte Form zu bringen [128]. Für die Ausführung von Prozessen verwenden sie dafür den Ansatz von [185]. Das vorgestellte Verfahren benötigt im ersten Schritt den zugrundeliegenden Smart Contract, welcher die Kollaboration enkodiert. Ähnlich zu [33], werden basierend auf den *function selector*, welcher die ersten vier Bytes einer gehashten Zeichenkettenrepräsentation der Funktionssignatur widerspiegelt, die entsprechenden ausgeführten Tasks identifiziert.

Überwachung

Neben der Prozessausführung wird die Blockchain-Technologie auch für eine Überwachung der Prozessausführung eingesetzt, wobei die Überwachung hier von der tatsächlichen Ausführungskomponente abgekoppelt ist.

Überwachung von Caterpillar

Di Ciccio et al.[↑] stellen in [33] ein Konzept zur nachvollziehbaren Überwachung der Prozessausführung mittels Caterpillar vor. Dabei werden für einen Abgleich mit den auf der Blockchain gespeicherten Daten die Keccak-Hashwerte für die Namen der Aufgaben des Prozessmodells manuell nachberechnet. Ähnlich zum Ansatz von Alves et al. [6], ist im Ansatz von Di Ciccio et al. die Ausführung Off-Chain, wobei sogenannte Oracles die Ereignisse in die Blockchain speichern, welche dann über eine *Trusted Layer* (deutsch: Vertrauensschicht) einer *Auditing Layer* (deutsch: Überprüfungsschicht) zur Verfügung stellt wird [34].

Bitcoin-Token zum Aufzeichnen der Historie

Prybila et al.[↑] liefern ebenfalls einen Beitrag zum Überwachen beziehungsweise Verifizieren des Prozessstatus und dessen Verlauf [143]. Dafür wird ein Bitcoin-Token verwendet, welches den aktuellen Stand reflektiert, ähnlich einem Token in einem Petri-Netz. Der aktuelle Besitzer des Tokens stellt den aktuellen Verantwortlichen beziehungsweise Ausführenden innerhalb der Kollaboration dar. Für parallelisierte Aufgaben, kann der Token gegebenenfalls auch geteilt und wieder zusammengeführt werden. Das Verfahren ermöglicht einerseits, dass bei Nichtbeachtung des Choreografie-Modells Strafen ausgesprochen werden können, oder andererseits, dass bei einer Teilnahme an der Kollaboration mögliche Entlohnungen eingefordert werden können. Bei diesem Ansatz handelt es sich also nicht um ein System zur Prozessausführung, sondern lediglich zum Überwachen, sodass auch nicht modellierte Situationen eintreten können.

Referenzarchitektur zum Überwachen

In [34] stellen **Di Ciccio et al.**[↑] eine Referenzarchitektur vor, wie die Überwachung der Prozessausführung im Blockchain-Umfeld konzipiert werden kann. Neben der Identifikation von zukünftigen Her-

ausforderungen im Forschungsbereich werden auch aktuelle Ansätze mit Bezug auf die vorgestellte Referenzarchitektur beziehungsweise auf die Herausforderungen analysiert.

Entscheidungsfindung

sichere, vertrauliche Entscheidungsfindung

Haarmann et al.[↑] beschäftigen sich in [64] mit dem Problem der Offenlegung sensitiver Daten bei der Ausführung von Entscheidungslogik. Der vorgestellte Ansatz erlaubt eine vertrauliche Entscheidungsfindung mittels DMN, wobei die zur Entscheidungsfindung beitragenden Variablen erst im Konfliktfall offengelegt werden.

Vertraulichkeit und Sicherheit

Kompromiss zwischen Vertraulichkeit und Enforceability

Köpke et al.[↑] diskutieren in [88] den Kompromiss zwischen *Privity* (deutsch: Vertraulichkeit) und *Enforceability* (deutsch: Vollstreckbarkeit) bei der Blockchain-basierten Prozessausführung. Statt die beiden Anforderungen als sich gegenseitig ausschließend zu betrachten, zeigen die Autoren, dass sich basierend auf sogenannten Sphären eine weiche Abstufung definieren lässt: Je mehr Teilnehmer sich in einer Sphäre befinden, desto geringer ist die Vertraulichkeit und desto besser ist die Vollstreckbarkeit. Für die praktische Umsetzung werden Entwurfsmuster zum Verschlüsseln und für den Schlüsselaustausch zwar evaluiert, allerdings nicht implementiert.

Kritische Analyse zur Anwendung von Blockchains

Die Arbeit von **Kumar et al.**[↑] beschäftigt sich mit einer tieferen Analyse der Anwendung der Blockchain-Technologie und deren Vor- und Nachteile innerhalb bestimmter Anwendungsbereiche [91]. Dafür werden unter anderem ein Metamodell für die Hyperledger-Blockchain aufgearbeitet, unterschiedlich konfigurierte Blockchains verglichen und Technologien zum herkömmlichen Datenaustausch mit der Blockchain-Technologie bezüglich bestimmter Kriterien gegenübergestellt. Dabei werden Vorteile bei der Blockchain bezüglich Netzwerktopologie, Integrität und Sicherheit von Daten und deren Verwaltung, Automatisierung sowie Vertrauen identifiziert. Als Nachteile in Zusammenhang mit der Blockchain-Technologie sind zum Teil deutlich höhere Kosten aufgrund der dezentralen Validierung und Replikation zu nennen. Zuletzt werden anhand von verschiedenen Anwendungsfällen die strategischen Entscheidungen bezüglich Vertrauen, Nachvollziehbarkeit, Sichtbarkeit und Kosten analysiert. Die Arbeit kommuniziert, dass eine unüberlegte Anwendung der Blockchain-Technologie Probleme wie etwa hohe Kosten verursachen könnte, welche sich in bestimmten Anwendungsfällen nicht amortisieren. Allerdings erfolgt die Betrachtung ohne Einbeziehung der Prinzipien der Prozessausführung (Kapitel 2.3).

Müller et al.[↑] analysieren vertrauenschaffende Eigenschaften von Blockchains im Prozesskontext [129]. Dafür erweitern sie ein beste-

hendes Konzept zur Beschreibung von Unsicherheit, Schwachstellen und Vertrauen und extrahieren daraus die folgenden fünf Dimensionen vertrauenssteigernder Funktionen für die Bildung der Taxonomie: (i) vertrauenssteigernde Methoden (zum Beispiel Unsicherheiten reduzieren), (ii) Quelle von Unsicherheiten (zum Beispiel die richtige Durchführung einer Aktivität), (iii) Vertrauenseigenschaften (zum Beispiel Integrität oder Verfügbarkeit), (iv) Prozesskomponenten (zum Beispiel organisationale Entitäten oder Daten) und (v) die Grenzen der Maßnahmen. Anhand dieser Dimensionen werden die folgenden Vertrauenseigenschaften von Blockchains im Prozessbereich evaluiert: Datenintegrität durch Hashing, Transparenz, Prozessinterpretation, automatisierte Ausführung von Aktivitäten durch Smart Contracts, Blockchain-basiertes Reputationssystem und Dezentralisierung. Damit wird zusammenfassend beschrieben, wie Blockchain-Systeme Vertrauen bei zwischenbetrieblichen Prozessen schaffen können.

Taxonomie für vertrauenssteigernder Eigenschaften der Blockchain

Evaluation spezifischer Anwendungsfälle

Fridgen et al.[↑] beschreiben unter Verwendung des DSR-Ansatzes, wie ein konkreter Anwendungsfall im Finanzsektor durch die Integration der Blockchain-Technologie optimiert werden kann [50]. Am Beispiel eines Akkreditivprozesses wird evaluiert, wie ein dokumentenlastiger Vorgang durch den Einsatz der Blockchain-Technologie digitalisiert und teilautomatisiert werden kann. In der Arbeit werden nicht die Prinzipien der Prozessausführung nach Kapitel 2.3 angewandt, sondern stattdessen im konkreten Fall die Vorteile der Blockchain gezeigt. Die gewonnenen Erkenntnisse deuten darauf hin, dass im Anwendungsfall die Blockchain-basierte Prozessausführung im engeren Sinne gewinnbringend eingesetzt werden kann.

Fallstudie aus dem Finanzsektor

Silva et al.[↑] stellen in [159] zwei Metamodelle vor, welche die Entwicklung von Blockchain-basierten Lösungen erleichtern sollen. Zum einen integrieren sie die Domäne der Hyperledger-Blockchain mit DEMO, einer Ontologie zur Modellierung von Geschäftstransaktionen in Unternehmen, und zum anderen wird ein gemeinsames Metamodell für Hyperledger und BPMN erstellt. Die Instanziierung der zwei Metamodelle wird für den Anwendungsfall einer Lieferkette in der Bananenindustrie demonstriert, um jeweils einen BPMN-Hyperledger Prototyp und einen DEMO-Hyperledger-Prototyp erstellen zu können. Die Evaluation fällt für beide Metamodelle aus funktionaler Sicht positiv aus, wobei die Verwendung des DEMO-Metamodells für eine schnellere Entwicklung einer Lösung auf der Blockchain sorgt. Der BPMN-fundierte Ansatz überzeugte hingegen in puncto Transaktionsmenge (85 im Gegensatz zu 174 beim DEMO-fundierten Ansatz).

DEMO-Hyperledger-Metamodell und BPMN-Hyperledger-Metamodell

8.5 SYSTEMATISCHE LITERATURANALYSEN

Kategorisierung unter anderem nach Prozesslebenszyklus, verwendete Techniken ...

Eine der ersten Literaturanalysen im Bereich Blockchain und BPM stellt die Arbeit von **Garcia-Garcia et al.**[↑] aus dem Jahr 2020 dar [54]. Unter anderem werden darin die folgenden Beobachtungen kommuniziert. In den analysierten Arbeiten liegt der Fokus bezüglich des Prozesslebenszyklus verstärkt auf der Blockchain-basierten Ausführung von Prozessen (50%) vor der Modellierung (34%) und der Analyse (16%). Die angewandten Forschungsmethoden verteilen sich annähernd gleichmäßig auf Proof of Concept-Implementierung (25%), Experiment (22%), Fallstudie (14%) und Arbeiten ohne Evaluation (35%). Auch die Anwendungsbereiche werden untersucht und ergeben einen starken Fokus auf universell einsetzbare Ansätze (58%), gefolgt von Lieferketten (20%) und den Forschungskontext (11%). Hinsichtlich technischer Charakteristika werden verstärkt die Ethereum-Blockchain und die BPMN-Modellierungssprache verwendet. Die Verteilung der Art der Veröffentlichungen beläuft sich auf 68% konzeptionelle Arbeiten, 59% Anwendungen und Werkzeuge sowie 32%, 26% und 15% für Framework, Methode oder (Meta)modell.

Anwendbarkeit Blockchain-basierter Prozessausführung im IoT-Bereich

Henry et al.[↑] analysieren in [69] Ansätze zur Dezentralisierung von Prozessausführungssystemen und berücksichtigten dabei explizit den Aspekt der IoT-Prinzipien. Herausforderungen wie die Benutzbarkeit, Vertraulichkeit oder Performance bremsen die breite Anwendung derartiger Systeme, wofür modellgetriebene Ansätze oder Blockchains mit beschränkter Teilnahme am Konsensverfahren Abhilfe schaffen sollen. In ihrer Analyse bewerten die Autoren verschiedene Ansätze hinsichtlich folgender Kriterien: empirische oder modellgetriebene [176, 185] Entwurfsmethoden, imperative [122] oder deklarative [115] Modellierungsparadigmen oder eine Prozess- [168] beziehungsweise Artefakt-zentrierte [160] Vorgehensweise. Auch Blockchain-spezifische Herausforderungen werden berücksichtigt, darunter Kostenoptimierung, Dauerhaftigkeit oder Interoperabilität. Neben den prozessbezogenen Kriterien Verwendbarkeit, Vertraulichkeit und Flexibilität werden darüber hinaus auch noch IoT-spezifische Kriterien mit einbezogen. Als Fazit werden offene Herausforderungen bezüglich der Flexibilität und Skalierbarkeit identifiziert. Explizit erwähnt werden zuletzt noch die Relevanz des Datenbezugs, so wie in [124] umgesetzt, und der Interoperabilität, zum Beispiel hinsichtlich der Unterstützung mehrerer Blockchains, so wie im BlockME2-Ansatz [46] umgesetzt.

Kategorisierung und Taxonomie bezüglich Prozessausführung

Die systematische Literaturrecherche von **Stiehle et al.**[↑] analysiert die Sicherheiten, Fähigkeiten und Potenziale (englisch: *guarantees and capabilities*) von Blockchains im Bereich des Prozessmanagements unter Entwicklung und Verwendung einer Taxonomie. Auch diese Arbeit identifiziert eine breite Verwendung des BPMN-Standards (36% Prozessdiagramme, 22% Choreografien und 11% Kollaborationsdia-

gramme) neben weiteren Alternativen wie YAWL, Petri-Netzen oder DCR-Graphen (insgesamt 29%). Die organisationale Perspektive, darunter meist rollenbasierte oder direkte Zuweisungen, wird von 58% der Ansätze integriert. Nur wenige Ansätze unterstützen flexiblere dynamische Zuweisungen. Hinsichtlich der Flexibilität der Prozessausführung, welche in Entkopplung, Adaption und Evolution untergliedert wird, können nur 13% der Ansätze überzeugen. Beim Blockchain-Protokoll entscheiden sich 67% für Ethereum, 25% unterstützen Hyperledger, während 11% eine andere, manuelle Implementierung verwenden. Am Ende identifizieren die Autoren in den Bereichen Flexibilität und Skalierbarkeit noch offene Herausforderungen, während die Nachvollziehbarkeit und Korrektheit der Ausführung in den analysierten Arbeiten bereits gut abgedeckt werden.

Kategorisierung nach ...

In dem ausführlichen Bericht von **Viriyasitavat et al.**[↑] werden existierende Ansätze anhand von verschiedenen Herausforderungen an zwischenbetriebliche Prozesse und bezüglich der Unterstützung bestimmter Charakteristiken von Prozessen und Stufen des Prozesslebenszyklus verglichen [178].

... Herausforderungen

Als Herausforderungen werden zum einen die Interoperabilität zwischen der Prozessausführung und der Blockchain-Domäne identifiziert, zum Beispiel, wenn externe Dienste auf verschiedenen Blockchains auf unterschiedliche Art und Weise implementiert werden müssen. Der BlockME-Ansatz [45, 46] von Falazi et al. adressiert diese Herausforderung. Zum anderen wird auch das durch die Blockchain entstehende Vertrauen in externe Dienste oder in die Prozessausführung selbst genannt, was von fast allen betrachteten Ansätzen umgesetzt wird. Als dritte Herausforderung wird die Transaktionsbestätigung beschrieben, insbesondere die fehlende Dauerhaftigkeit aufgrund von Blockchain-Forks, was ebenfalls von Falazi et al. [45, 46] betrachtet wird. Der letzte Punkt bezieht sich auf die Flexibilität der Prozessausführung, welche aufgrund der Irreversibilität der bereitgestellten Smart Contracts explizit implementiert werden muss. Dies ist speziell in den Arbeiten von López-Pintado et al. [107] und Klinger et al. [84, 85] umgesetzt.

... Prozesscharakteristika

Bei den Prozesscharakteristika werden transiente, also kurze und kurzfristige Kollaborationen, welche lediglich von López-Pintado et al. [107, 109] und Klinger et al. [84] unterstützt werden, den persistenten, langfristigen Kollaborationen gegenübergestellt, welche vom Großteil der betrachteten Arbeiten unterstützt werden. Ebenso werden dynamische Kollaborationen mit Änderungen am Prozess und in den aufgerufenen Diensten den statischen Kollaborationen gegenübergestellt. Dabei fokussieren sich López-Pintado et al. [107–109] sowie Klinger et al. [84, 85] auf dynamische Kollaborationen im Gegensatz zu unter anderem Weber et al. [185], Sturm et al. [170], Nakamura et al. [134] oder López-Pintado et al. [110, 113]. Der Bezug zum Prozesslebenszyklus wird auch in den Charakteristiken mit auf-

geführt, denn manche Ansätze fokussieren die Modellierung eines Blockchain-gesicherten Verzeichnis, in dem die Modelle sicher gespeichert werden, während sich andere Ansätze auf eine sichere Ausführung konzentrieren. Diesbezüglich werden Sturm et al. [170], Klinger et al. [84] oder Falazi et al. [46] (BlockME2) in den Bereich Modellierung eingeordnet, während Weber et al.[185] und López-Pintado et al. [113] in die Kategorie Ausführung fallen. Falazi et al. [45] (BlockME) wird in keine dieser Kategorien eingeordnet. Zuletzt wird unterschieden, ob eine private Blockchain für zentralisierte Prozesse oder eine öffentliche Blockchain für verteilte, dezentrale Prozesse unterstützt wird. Dabei sind die meisten Ansätze auf den Einsatz in dezentralen Strukturen ausgerichtet.

... Prozesslebenszyklus

Die Einordnung in den Prozesslebenszyklus erfolgt nach den Stufen Modellierung, Konfiguration oder Ressourcenallokation sowie Ausführung und schließlich Analyse oder Diagnose. Die Arbeit von García-Bañuelos et al. [53] wird ausschließlich in die Ausführungsphase eingeordnet, ebenso wie das Caterpillar-System [110, 113]. In die Analysephase werden Ladleif et al. [96] (auch Konfiguration) sowie Ladleif et al. [107, 109], Klinger et al. [84], Klinkmüller et al. [86], Mühlberger et al. [128], Madsen et al. [115] (alle auch Ausführung) eingeordnet. Prybila et al. [143], Di Ciccio et al. [33], Haarmann et al. [63] und Silva et al. [159] werden ausschließlich in die Phase Analyse eingeordnet. Arbeiten, welche sowohl Modellierung, Konfiguration beziehungsweise Ressourcenallokation und Ausführung behandeln, sind Weber et al. [185], Sturm et al. [170] und Klinger et al. [85].

Für die Einordnung mancher Arbeiten sind in der Arbeit nur oberflächlich die Entscheidungsgründe diskutiert. Beispielsweise werden die Publikationen der in Kapitel 5 vorgestellten Arbeit [170] einerseits ausschließlich in die Modellierungsphase eingeordnet, obwohl darin Ausführungskonzepte im Vordergrund stehen, und andererseits wird die Erweiterung dazu hinsichtlich der organisationalen Perspektive [168] später dann trotzdem auch in die Ausführungsphase eingeordnet. Weiterhin sind in der Analysetabelle manche Arbeiten wie [108] mehrmals in den Zeilen aufgelistet während andere Arbeiten gar nicht betrachtet sind, zum Beispiel ChorChain von Corradini et al. [27, 29]. Die Qualität der Studie soll hier nicht weiter diskutiert werden.

Analyse

Curty et al.[↑] vergleichen in ihrer Arbeit Ansätze aus dem akademischen Bereich der modellgetriebenen Softwareentwicklung und darüber hinaus auch Low-Code und No-Code-Plattformen aus dem industriellen Umfeld [30]. Basierend auf einer dreidimensionalen Klassifikation werden die insgesamt 232 betrachteten akademischen Ansätze hinsichtlich ihrer Modellierungssprache, der Blockchain-Technologie sowie der *Nature of realization* (deutsch: Realisierungsstrategie) quantitativ analysiert. Innerhalb der Modellierungssprache verwenden 32% der Ansätze eine domänenspezifische Sprache, gefolgt von

BPMN (22%), UML (16%) und Petri-Netzen (4%). Ethereum ist die am häufigsten verwendete Blockchain-Technologie (43%). 17% arbeiten unabhängig eines bestimmten Protokolls, 7% unterstützen mehrere Protokolle und 6% verwenden die Hyperledger-Blockchain. Bei der Realisierungsstrategie werden konzeptuelle Ansätze und Codegenerierende Ansätze unterschieden. Erstere stützen sich auf die Entwicklung einer Modellrepräsentation auf der Blockchain, worauf Analysen und Schlussfolgerungen gezogen werden (interpretierender Ansatz). Zweitere nutzen eine gewisse Eingabeinformation, womit automatisch ausführbarer Quelltext generiert wird, darunter Smart Contracts oder Programmcode für grafische Benutzerschnittstellen (kompilierender Ansatz).

Automatische Inhaltsanalyse

Auf Basis einer automatisierten Inhaltsanalyse[4] der akademischen Veröffentlichungen werden acht Themen abgeleitet, in welche die Ansätze nach manuellen Verfeinerungsschritten und Diskussionsrunden eingeordnet werden. Dabei werden 48 Arbeiten dem Thema *Process, workflow, choreography, and decision models* zugeordnet, 42 Arbeiten behandeln das Thema Anwendungsentwicklung (*Application development*) und 15 Arbeiten fokussieren sich auf formale Verifikationen. Die restlichen 30 betrachteten Arbeiten teilen sich auf die Themen Ontologien, UML, Geschäftsprozessmodellierung, Referenzmodelle und Zuliefererketten auf.

Fazit

Als Fazit erkennen die Autoren, dass die akademischen Ansätze größeren Wert auf Modellierung und Konzeptionalisierung legen, während die industriellen Plattformen mit technischer Ausgereiftheit überzeugen können. Weiterhin wird die Wichtigkeit der implementierungstechnischen Umsetzung von Konzepten als essenziell herausgestellt, etwa die Generierung von Quelltext der kompilierenden Ansätze. Auch wird hinsichtlich der Modellierung der Bedarf an Blockchain-orientierten Ansätzen verdeutlicht, so wie von SecBPMN2BC [92] umgesetzt. Zuletzt wird vorgeschlagen, akademische Ansätze über Plug-in-Mechanismen in existierenden Plattformen verfügbar zu machen oder Funktionalitäten über definierte APIs zu integrieren. Auch bei dieser Arbeit zeigt sich ähnlich zu Viriyasitavat et al. [178] eine nicht nachvollziehbare Einordnung mancher Ansätze. Beispielsweise wird Sturm et al. [169] zu den BPMN-Choreografie-Ansätzen zugeordnet [30, Kap. 6], obwohl der Ansatz auf BPMN-Prozessdiagrammen aufbaut und die Choreografien lediglich zur Abgrenzung von anderen Arbeiten verwendet.

Lichtenstein et al.[↑] beschränken den Fokus ihrer Literaturanalyse in [103] auf die Unterstützung von Flexibilität im Rahmen der Blockchain-basierten Prozessausführung, insbesondere in Bezug auf die verhaltensorientierte aber auch auf die organisationale Perspektive. Eine flexible Ausführung ist bei der Prozessausführung entscheidend, da das dynamische Verhalten während der Laufzeit zum Zeit-

4 nach dem Latent Dirichlet Allocation-Verfahren

punkt der Bereitstellung oft noch nicht vollständig antizipiert werden kann. Die kompilierenden Ansätze haben Schwierigkeiten bei der Umsetzung der Flexibilität, da die Prozesslogik als Smart Contract-Quelltext bereitgestellt wird, welcher nach den Prinzipien der Blockchain nach der Bereitstellung nicht mehr geändert werden kann. Die interpretierenden Ansätze verwalten das Prozessmodell hingegen in einer Datenstruktur innerhalb eines Smart Contracts und können dadurch mittels weiterer Transaktionen die Datenstruktur während der Ausführung anpassen. Die Literaturanalyse beschäftigt sich einerseits mit den Fragen nach der Unterstützung von Flexibilität aktueller Ansätze und identifiziert andererseits noch offene Forschungsfragen in dieser Hinsicht.

Analyse: Umsetzung der Flexibilität

Die Analyse gliedert sich nach den Dimensionen flexibler Prozessausführung, welche aus bestehender Literatur abgeleitet werden. Die Ergebnisse werden im Folgenden zusammengefasst. Die Flexibilität ist generell stärker in der organisationalen Perspektive umgesetzt als in der verhaltensorientierten Perspektive, wobei in letzerer verstärkt ein iteratives Planungsverfahren unterstützt wird [16, 107, 115, 193, 197], das heißt, die finale Spezifikation erfolgt zur Laufzeit. Bei der Umsetzung der Flexibilität innerhalb der organisationalen Perspektive hingegen erfolgt ungefähr bei der Hälfte der Ansätze eine dynamische Konfiguration des Prozesses vor der Ausführungsphase [16, 85, 96, 112, 115, 168, 170, 185] und bei der anderen Hälfte ebenfalls während der Ausführung [28, 29, 84, 107–109, 111]. Das Ausmaß der Flexibilität ist bei der verhaltensorientierten Perspektive teilweise nicht beschränkt [16, 115, 192] beziehungsweise teilweise regional im Prozessmodell beschränkt [107, 197]. Bei der organisationalen Perspektive liegt das Verhältnis stark zugunsten der uneingeschränkten Flexibilität [28, 29, 84, 85, 96, 107–109, 111, 112, 115, 168, 170, 185]. Der Automatisierungsgrad ist bei der verhaltensorientierten Perspektive höher als bei der organisationalen Perspektive, wo die Flexibilität eher manuell umgesetzt wird. Zuletzt wird die Entscheidungsfindung untersucht, mit dem Ergebnis, dass bezüglich der verhaltensorientierten Perspektive die Spezifikation eher regelbasiert erfolgt (nur bei [197] Endanwender getrieben), während bei der organisationalen Perspektive eine Endanwender-getriebene Spezifizierung bevorzugt umgesetzt wird (nur im Caterpillar-System [107–109] regelbasiert).

offene Herausforderungen

Als offene Forschungsherausforderungen beschreiben die Autoren eine stärkere Unterstützung des Ausspezifizierungsprozesses, eine Kostenoptimierung der Ansätze und die Berücksichtigung der Flexibilität sowohl bei der Auswahl der Modellierungssprachen als auch bei den Implementierungsparadigmen. Das dpex-Framework adressiert die Kosten durch den Einsatz von Thin Contracts, die Flexibilität hinsichtlich der Modellierungssprachen ist durch die Anbindung einer entsprechenden BPM-Komponente möglich und die Komplexität

hinsichtlich der Implementierung ist durch die Unabhängigkeit von der Blockchain auch als gering einzuschätzen.

In der Literaturanalyse [177] evaluieren die Autoren **Viriyasitavat et al.**[↑] die selektierten Arbeiten einerseits nach den Herausforderungen bezüglich der Blockchain-Technologie und BPM, nämlich Interoperabilität, Vertrauen/Sicherheit, Finalität und Flexibilität beziehungsweise der Reversibilität von Blockchains. Andererseits erfolgt die Evaluation anhand von *Business Process Compliance*-Kriterien, welche als technische, organisationale, rechtliche, soziale und ökonomische Herausforderungen kategorisiert werden.

Resultate der Arbeit

Grundsätzlich werden die bezüglich dieser Arbeit wichtigsten Resultate in diesem Kapitel bereits durch andere Publikation wiedergegeben. Die Interoperabilität von Blockchains ist wichtig, sodass Daten leicht ausgetauscht und externe Systeme angebunden werden können. Das dpex-Framework unterstützt dies durch die Trennung von BPM und SCI. Vertrauen ist ein entscheidender Punkt bei der Blockchain-basierten Prozessausführung und bedarf einer anwendungsspezifischen Untersuchung. In dpex ist es unter anderem durch die Schnittstellendefinition möglich, unkompliziert neue Blockchain-Protokolle oder Blockchain-Konfigurationen anzubinden. Auch das Problem der probabilistischen Finalität wird angesprochen, was bei Falazi et al. [45, 46] diskutiert wird und durch die Erweiterung des Ethereum-Adapters auch in dpex adressiert werden kann.

Grenzen der Arbeit

Ähnlich zu einer früheren Literaturanalyse derselben Autoren [178] konnten auch hier nicht alle Einordnungen der Publikationen in die Kategorien nachvollzogen werden. Zum Beispiel ist nach [177] die Standardisierung zur einfacheren Integration von Blockchain-basierten Diensten in die Prozessausführung lediglich bei Sturm et al. [170] und García-Bañuelos et al. [53] umgesetzt und bei Weber et al. [185], Caterpillar oder Lorikeet nicht, obwohl die Unterschiede dieser Arbeiten in Bezug auf die Interoperabilität nicht eindeutig kommuniziert sind. Weiterhin sei die Herausforderung bezüglich des Vertrauens oder der Sicherheit von allen betrachteten Ansätzen umgesetzt, jedoch im Ansatz von Madsen et al. [115] nur partiell, wofür auch keine klare Begründung oder Analyse angegeben ist. Bezüglich der probabilistischen Finalität ist zwar die Arbeit von Falazi et al. [45, 46] im Text referenziert, allerdings nicht in der Ergebnistabelle eingeordnet. In der Tabelle wird die Berücksichtigung der Finalität lediglich den Systemen Caterpillar und Lorikeet zugeschrieben. Allerdings sind auch hier keine speziellen Komponenten hervorgehoben, wodurch sich diese Systeme in dieser Hinsicht von den anderen Ansätzen wie zum Beispiel von Weber et al., García-Bañuelos et al. [53], Nakamura et al. [134], Sturm et al. [170] und dem BPMChain-System von Schinle et al. [152] unterscheiden. Die letzte Herausforderung betrifft die Flexibilität in der Prozessausführung (englisch: *Reversibility*). Diesbezüglich wird im Fließtext bemerkt, dass die Ansätze von Ma-

dsen et al. [115], López-Pintado et al. [109] und Klinger et al. [84, 85] Flexibilität berücksichtigen, aber bislang nicht vollständig umsetzen. In der finalen Ergebnistabelle tauchen diese Resultate hingegen nicht auf.

8.6 ARBEITEN MIT ENTFERNTEM BEZUG ZUM FORSCHUNGSBEITRAG

Idelberger et al.[↑] stellen in [74] das imperativ-prozedurale Implementierungsparadigma den deklarativen Ansätzen zur Erstellung von Smart Contracts gegenüber. Obwohl dies zuerst unabhängig vom Forschungsbereich des Geschäftsprozessmanagements ist, existieren trotzdem Berührungspunkte. Zum Beispiel zeigen die Autoren Nachteile bei der manuellen imperativen Programmierung auf, welchen durch den Einsatz von Zustandsautomaten entgegnet werden können. Diese können als Unterstützung zur automatischen Kompilierung von Smart Contracts, ähnlich zu [185], oder zur direkten Ausführung, ähnlich zu [134, 170], verwendet werden.

Bore et al.[↑] stellen ein Werkzeug zur Verwaltung und zum Benutzen von Prozessen auf Blockchain-basierten Lösungen vor [14]. In der Arbeit wird ein *Web Dashboard* hervorgehoben, mit dem Prozesse erstellt werden können. Diese Prozessbeschreibungen werden dann benutzt, um automatisiert die entsprechenden Elemente in einer grafischen Benutzerschnittstelle zu erstellen. Die automatisierte Generierung von Smart Contracts auf Basis der Prozessbeschreibungen ist in dem Ansatz bislang nicht implementiert und soll in künftigen Arbeiten integriert werden. Genaue Lösungsansätze dafür werden nicht beschrieben, die Idee jedoch ist den Anmerkungen zufolge unkonventionell, denn die automatisierte Generierung von Chaincode soll auf Basis von Techniken aus dem Bereich des maschinellen Lernens, zum Beispiel rekurrenten neuronalen Netzen, konzipiert werden.

Die Arbeit von **Meroni et al.**[↑] liefert einen Beitrag im Bereich der Prozessüberwachung [124]. Darin wird ein System vorgestellt, mit dem Prozesse, welche auf einer Blockchain ausgeführt werden, in einer Artefakt-zentrierten Art und Weise überwacht werden. Das bedeutet, dass zum Beispiel die Information über das Beenden einer Aktivität *Tank an Lkw anhängen* anhand des Artefakts *Tank* und dessen beiden Status *gefüllt* und *angehängt* abgeleitet werden kann. Im Gegensatz zu anderen Ansätzen, kann dadurch Verhalten überwacht und erkannt werden, welches vom ursprünglichen Prozessmodell abweicht. Dies ist explizit als Anforderung an das System mit aufgeführt. Als Nachteil beschreiben die Autoren, dass kein Kontrollmechanismus integriert ist, welcher die Überwachungsdaten auf Korrektheit überprüft, bevor diese in die Blockchain geschrieben werden.

Die Literaturrecherche von **Chang et al.**[↑] berücksichtigt in der Analyse ausschließlich Ansätze mit Bezug zum Supply Chain Ma-

nagement [24]. Arbeiten aus dem BPM-Forschungsbereich werden lediglich für Hintergrundinformation zu Blockchain oder Ähnlichem verwendet.

In ihrer Dissertation [80] entwirft **Barkha Javed**[↑] ein Rahmenwerk zur Nachvollziehbarkeit politischer Entscheidungen. Die Arbeit von Sturm et al. [168] und ähnliche Arbeiten werden zwar als Möglichkeit erwähnt, die Herkunft von Daten oder Entscheidungen nachzuvollziehen, jedoch werden Blockchain-basierte Lösungen in der Arbeit generell nicht weiter verfolgt.

In der Arbeit [87] liefern **Köpke et al.**[↑] keinen neuen wissenschaftlichen Beitrag, sondern rekapitulieren lediglich die Kernpunkte von [88].

Ndadji et al.[↑] stellen basierend auf attributierten Grammatiken einen neuen, generellen Ansatz zur Prozessmodellierung vor [196], welcher für administrative Prozesse spezialisiert sein soll und sowohl den Lebenszyklus als auch die informationsorientierte und organisationale Perspektive abdecken soll. Der Ansatz selbst ist nicht explizit durch die zwischenbetrieblichen Prozesse motiviert, jedoch wird die stärker werdende Bedeutung der Blockchain in der Prozessausführung angemerkt und die Unterstützung verteilt ausgeführter Prozesse durch den vorgestellten Ansatz.

Bodziony et al.[↑] stellen ein Wallet vor, welches basierend auf Smart Contracts Blockchain-Adressen miteinander verknüpfen kann, um die Gefahr von Fehlern durch die Verwendung von den ursprünglichen Wallet-Adressen zu verringern [13]. Die Arbeit liegt nicht im Bereich des Geschäftsprozessmanagements, jedoch wird die Blockchain-basierte Prozessausführung als eine mögliche Anwendung für den vorgestellten Ansatz angemerkt.

Der Fokus von **Braun et al.**[↑] in [17] liegt auf der Überwachung und verifizierbaren Dokumentation eines Prozesses. Für Prozessinstanzen können zusätzlich ausgewählte Daten mit unterschiedlichen Vertraulichkeitsanforderungen mit anderen Organisationen geteilt werden, sodass diese ebenfalls über die Blockchain gesichert sind. Für die Repräsentation der Prozesse wird die WiLD-Ontologie verwendet, was eine Auswahl von Teilnehmern und Sub-Prozessen zur Laufzeit erlaubt. Die Blockchain beziehungsweise Distributed Ledger-Technologie wird jedoch hier nur zu Dokumentationszwecken verwendet und nicht zur Prozessausführung, welche hier außerhalb der Blockchain erfolgt.

Kowalski et al.[↑] veröffentlichen mit [89] Interviews mit Experten aus dem Industriesektor der Handelsfinanzierung und deren Einschätzungen bezüglich des vertrauenschaffenden Einflusses der Blockchain. Es fehlt hier der Bezug zum Prozessmanagement, wobei der Artikel von Sturm et al. [168] lediglich als breitere Anwendung der Blockchain über verschiedene Industriezweige hinweg erwähnt wird.

In [62] verwenden **Haarmann et al.**[↑] ein zeitbehaftetes Petri-Netz, um die zeitlichen Auswirkungen der Blockchain auf die Prozessausführung zu simulieren. Dabei kommen die Autoren zu dem Schluss, dass sich der zeitliche Mehraufwand bei der Blockchain-basierten Ausführung von häufigen und hochautomatisierten Prozessen deutlich stärker auswirkt als bei weniger häufig ausgeführten Prozessen mit überwiegend manuellen Schritten.

Henry et al.[↑] entwickeln ein System hinsichtlich der organisationalen Perspektive [68], ähnlich zu [108]. Die Autoren entwickeln ein Protokoll, nach dem sich mögliche Prozessteilnehmer in einem Smart Contract registrieren können, welcher zur Ausführungszeit basierend auf den Anforderungen und Filterkriterien die passende Ressource automatisch auswählt und zuweist. Dies ist aktuell nicht direkt in die Prozessausführung integriert, jedoch soll die Anbindung eines WFMS leicht möglich sein.

In [67] veröffentlichen **Henry et al.**[↑] die Ergebnisse ihrer DSR-Studie zur Entwicklung eines dezentralen Systems zur Verwaltung von Logistikdienstleistungen auf Basis der Blockchain-Technologie. Ähnlich wie Fridgen et al. [50] werden nicht explizit die allgemein anwendbaren Prinzipien der Prozessausführung integriert, sondern stattdessen ein spezielles Fallbeispiel betrachtet.

In ihrer Dissertation [66] forscht **Tiphaine Henry**[↑] im Bereich der zwischenbetrieblichen Prozessausführung. Darin werden insbesondere die Herausforderungen der Public-Private Prozesse, Laufzeitflexibilität sowie die Vertraulichkeit von Daten beschrieben.

In [82] stellen **Kaiya et al.**[↑] eine auf Konformitätsprüfung beruhende Methode vor, um korrelierende Aktivitäten verschiedener Prozesse zu finden, welche zur Ausführung der jeweils anderen Aktivität beitragen könnten. Der Ansatz ist weder auf zwischenbetriebliche Prozesse fokussiert, noch wird die Blockchain-Technologie in jeglicher Art und Weise mit einbezogen. Sturm et al. [169] wird als verwandte Arbeit erwähnt, welche sich jedoch auf die Ausführung und Überwachung zwischenbetrieblicher Prozesse fokussiert.

In der Arbeit [156] diskutieren **Sedlmeir et al.**[↑] die Herausforderungen in Bezug auf die für die Aufrechterhaltung der Funktionalität notwendige Transparenz von Transaktionen in einem Blockchain-Netzwerk und die daraus entstehenden Probleme hinsichtlich Frontrunning oder der Datenschutzgrundverordnung. Als Lösungsansätze werden sogenannte *permissioned* Blockchains, *Self-Sovereign Identities* oder *Zero-Knowledge Proofs* diskutiert.

Sali et al.[↑] elaborieren in [149] allgemein die Vorteile der Blockchain bezüglich Transparenz und Nachvollziehbarkeit von Lieferketten im Bereich der Nahrungsmittelindustrie, ohne dabei auf die Prinzipien des Geschäftsprozessmanagements einzugehen.

In [163] stellen **Stiehle et al.**[↑] ihre Forschung vor, welche die Skalierbarkeit Blockchain-basierter Prozessausführung durch den Einsatz

von sogenannten *Channels* verbessern soll. Die Arbeit ist noch in einem frühen Stadium, wobei erste Resultate vielversprechend in Bezug auf Durchsatz und Kostenreduktion sind.

Suliman et al.[↑] stellen in [171] das BlockCheck-Rahmenwerk vor. BlockCheck ist kein WFMS im Bereich der Prozessausführung, sondern dient dazu, basierend auf einem Prozessmodell, die Konformität auftretender Ereignisse zu überprüfen. Dafür werden Regeln aus einem BPMN-Prozessdiagramm extrahiert. Diese Regeln beziehen sich einerseits auf den Kontrollfluss durch die Übersetzung in ein Petri-Netz und andererseits werden Regeln für komplexe BPMN-Konstrukte wie Events oder zeitliche sowie organisationale Bedingungen extrahiert. Bei Aktivitäten werden dabei sowohl das Starten als auch das Beenden einer Aktivität berücksichtigt.

Swain et al.[↑] stellen einen Ansatz vor, die Blockchain im Bereich von Lieferketten einzusetzen [172]. Dabei werden aber keine allgemeinen Prinzipien der Prozessausführung angewandt.

Die Autoren **Xu et al.**[↑] schlagen in [193] einen Fragmentierungsansatz zur Realisierung von Nutzerprivilegien und Zugriffskontrolle auf der Blockchain vor. Der Ansatz liegt demnach auch außerhalb des Forschungsbereichs der Prozessausführung.

8.7 SONSTIGE ARBEITEN

Folgende Publikationen sind unter anderem aufgrund offensichtlich falscher Zitierungen aussortiert.

Wang et al.[↑] [184] verwendet Sturm et al. [168] als Referenz für den Miller-Rabin Algorithmus zur Primzahlenerkennung.

Farrugia et al.[↑] [48] verwendet Sturm et al. [168] als Referenz für die Blockchain-interne Verarbeitung von Transaktionen.

Yu et al.[↑] [194] verfasste eine nicht englischsprachige und nicht deutschsprachige Publikation.

Wan et al.[↑] [182] verwendet [168] als Referenz für den Forschungsbereich der Sicherheitsanalyse von Smart Contracts.

Rikken et al.[↑] analysieren in [145] dezentralisierte autonome Organisationen. Die Arbeit hat somit keinen Bezug zum Prozessmanagement. Die Veröffentlichung [168] wird lediglich als Grundlagenliteratur zu Blockchain-Systemen referenziert.

9

CONCLUSIO UND ANSCHLIESSENDE FORSCHUNGSARBEIT

Der letzte Abschnitt fasst die Inhalte dieser Forschungsarbeit in Kapitel 9.1 zusammen und listet in Kapitel 9.2 die Beiträge noch einmal übersichtlich auf. Abschließend werden in Kapitel 9.3 die Limitationen der aktuellen Ansätze dargestellt und Anknüpfpunkte für künftige Forschungsarbeiten aufgezeigt.

9.1 ZUSAMMENFASSUNG

Diese Forschungsarbeit beschäftigt sich mit der dezentralen Verwaltung von Prozesskontrolldaten und liefert damit einen Beitrag zur dezentralen Ausführung von Prozessen im Forschungsbereich des Geschäftsprozessmanagements.

Teil I: Grundlagen BPM und zwischenbetriebliches Prozessmanagement

In dieser Arbeit werden zuerst die Grundlagen des Geschäftsprozessmanagements eingeführt, speziell mit Fokus auf die Modellierung und WFMS-gestützte Ausführung von Prozessmodellen. Insbesondere werden die verschiedenen Perspektiven von Prozessmodellen, darunter die organisationale Perspektive für die Spezifikation von Verantwortlichen für Aktivitäten auf Basis eines Organisationsmodells, die informationsorientierte Perspektive zur Integration von Daten oder die operationale Perspektive zur Anbindung von externen Anwendungen und Programmen dargestellt. Anschließend wird deren Umsetzbarkeit mit BPMN und Camunda in Hinblick auf die entwickelten Beispielkonnektoren für dpex demonstriert. Prozessausführungssysteme können auf verschiedenen IT- und Netzwerkinfrastrukturen bereitgestellt werden, welche zentral verwaltet oder verteilt organisiert sind oder auf dezentralen Peer-to-Peer-Netzwerken basieren. Darauf können jeweils Prozesse, vor allem im zwischenbetrieblichen Umfeld, mit unterschiedlichen Paradigmen auf Basis einer prozessbasierten oder nachrichtenbasierten Modellierung umgesetzt werden. Für die Abgrenzung der in dieser Arbeit fokussierten dezentralen Prozessausführung werden die unterschiedlichen Kombinationen aus Modellierungsparadigma und implementierungstechnischer Umsetzung dargestellt und unter Berücksichtigung des Sicherheitsaspekts diskutiert. Dabei werden auch die verschiedenen Formen von

Workflow-Interoperabilität vorgestellt und um die Form von Decentralized Control erweitert.

Teil II: Blockchain-basierte Prozessausführung

Blockchain-basierte Systeme haben sich als geeignete Infrastruktur für die Implementierung der dezentralen Prozessausführung erwiesen. In dieser Arbeit werden die Grundlagen und die Funktionsweise von Blockchain-Protokollen eingeführt, bevor darauf aufbauend die Durchführung eines ersten DSR-Zyklus beschrieben wird. Darin wird ausgehend vom existierenden Konzept des kompilierenden Ansatzes bei der Blockchain-basierten Prozessausführung, bei der ein gegebenes Prozessmodell in einen modellspezifischen Smart Contract-Quelltext übersetzt wird, ein flexibler interpretierender Ansatz vorgeschlagen, um eine WFMS-ähnliche Prozessausführung durch die Interpretation eines Prozessmodells zur Laufzeit auf der Ethereum-Blockchain zu ermöglichen.

Teil III: dpex

Die im DSR-Evaluationsschritt neu identifizierten Probleme, unter anderem die Notwendigkeit der Reimplementierung von WFMS-Funktionalitäten in Smart Contracts, begründet durch eine fehlende Schichtentrennung von Prozessausführung und Blockchain-Systemen, werden daraufhin in einem zweiten DSR-Zyklus aufgearbeitet. Die Lösung der Herausforderungen erfolgt über dpex, einem Rahmenwerk zur dezentralen Prozessausführung. Das dpex-Framework forciert eine klare Schichtentrennung und integriert existierende WFMSs zur Prozessausführung zusammen mit SCIs zur sicheren Nachrichtensynchronisierung. Dadurch lassen sich verschiedene WFMSs und SCIs über Adapterimplementierungen anbinden und flexibel und unabhängig voneinander für die kollaborative Ausführung von zwischenbetrieblichen Prozessen konfigurieren. Im praktischen Einsatz verwalten dann verschiedene Organisationen jeweils sowohl eine lokale dpex-Instanz, ein WFMS wie Camunda, welches den zwischenbetrieblichen Prozess interpretieren kann und einen lokalen Knoten einer SCI wie der Ethereum-Blockchain. Damit werden alle Prozessschritte kollaborationsweit koordiniert, indem alle Benutzer über die dpex-Benutzerschnittstelle ihre personalisierte Arbeitsliste einsehen, die Ausführungsrechte von Aktivitäten beanspruchen und die Bearbeitung darüber auch abschließen können. Eine SCI, wie etwa Ethereum, sorgt dabei für eine eindeutige Reihenfolge aller Ereignisse, sodass konkurrierende Anfragen automatisch aufgelöst werden, während die organisationsinternen WFMS-Instanzen, wie etwa Camunda, die Prozesskonformität aller Ereignisse prüfen und den gesamten Prozessstatus damit dezentral verwalten.

Teil IV: Verwandte Arbeiten und bewertende Schlussfolgerungen

Weiterhin bietet diese Arbeit einen umfassenden Überblick über die relevantesten Veröffentlichungen im Bereich der Blockchain-basierten Prozessausführung, welche mittels der Schneeballmethode für systematische Literatur-Übersichtsarbeiten selektiert werden. Die ausgewählten Artikel und Beiträge werden zuerst der Relevanz nach sortiert. Anschließend werden die Ansätze im Kern des Forschungsbe-

reichs der Blockchain-basierten Prozessausführung weiterhin anhand von bestimmten Kriterien kategorisiert, wie etwa nach der verwendeten Modellierungssprache, dem unterstützten Blockchain-Protokoll oder hinsichtlich Flexibilitätskriterien. Außerdem werden weitere systematische Literatur-Übersichtsarbeiten und deren Erkenntnisse vorgestellt. In diesem letzten Kapitel erfolgt schließlich eine bewertende Zusammenfassung dieser Arbeit, wobei mögliche Anknüpfpunkte für zukünftige Forschungsarbeiten aufgezeigt werden.

9.2 BEITRAG DIESER FORSCHUNGSARBEIT

Dieses Kapitel fasst den Gesamtbeitrag dieser Forschungsarbeit zusammen. Dafür wird zuerst das Forschungsgebiet in Kapitel 9.2.1 abgegrenzt, bevor der Stand der Wissenschaft vor dieser Arbeit noch einmal in Kapitel 9.2.2 beschrieben wird und schließlich die einzelnen Beiträge in Kapitel 9.2.3 rekapituliert werden.

9.2.1 *Abgrenzung des Forschungsgebiets*

Das Kernthema dieser Forschungsarbeit ist je nach technischer oder organisatorischer Sichtweise die dezentrale oder zwischenbetriebliche Prozessausführung. In der akademischen Welt wird oftmals nicht zwischen diesen beiden Perspektiven unterschieden und die Begriffe werden unterschiedlich verwendet. Aus diesem Grund sollen die weiteren Forschungsgebiete kurz abgegrenzt werden, bevor der Beitrag dieser Arbeit beschrieben wird. Der folgende Abschnitt hat keine umfassende Analyse der Terminologie in der Literatur zum Ziel, sondern zeigt exemplarisch die unterschiedliche Sichtweise auf die dezentrale und zwischenbetriebliche Prozessausführung.

Anforderungen an zwischenbetriebliche Prozesse

Die Anforderungen an zwischenbetriebliche Prozesse decken sich größtenteils mit den in dieser Arbeit zugeschriebenen Eigenschaften der dezentralen Prozessausführung. Die Anforderungen umfassen unter anderem den Erhalt der Autonomie von beteiligten Organisationen sowie die Unterstützung heterogener Umgebungen und Systemlandschaften [105] sowie die Möglichkeit der Trennung zwischen privaten und zwischenbetrieblichen Prozessen [106]. Je nach der gewählten Umsetzungsstrategie werden darüber hinaus in der Literatur noch spezifische Anforderungen definiert. Dazu zählen das automatisierte Entdecken von Geschäftspartnern und Anbietern von Diensten bei den Webservice-getriebenen Loosely Coupled Workflows (siehe unten) oder aber die Integration menschlicher Agenten oder die globale Nachverfolgung des Prozessfortschritts [106].

Die Umsetzung zwischenbetrieblicher Prozesse lässt sich grob in zwei Kategorien einteilen, nämlich in lose gekoppelte Geschäftsprozesse (Loosely Coupled Workflows, Kapitel 3.5) und in die tatsächliche Interpretation und Ausführung eines global definierten zwischen-

betrieblichen Prozessmodells nach WFMS-Prinzipien (Kapitel 2.3 und Kapitel 2.4).

zwischenbetriebliche Prozessausführung nach LCW

Die Umsetzung des Loosely Coupled Workflows-Prinzip kann wiederum auf unterschiedliche Weise erfolgen. Eine Möglichkeit stellt die auf serviceorientierten Architekturen basierte und mit Webservice-Spezifikationen beschriebene Integration von verteilt ausgeführten Geschäftsprozessen zur automatisierten Abarbeitung von Teilprozessen dar [187, Kap. 8]. Eine andere Möglichkeit beschreibt das Austauschen von Nachrichtenobjekten, etwa über das HTTP-Protokoll, zwischen zwei auf verschiedenen WFMS ausgeführten Prozessen, so wie in Kapitel 3.2 dargestellt. In beiden Fällen muss unter anderem auf die Gefahr von Deadlocks bei der Spezifikation der Nachrichtenabfolge oder auf die Interoperabilität hinsichtlich des Datenaustausches geachtet werden [2, 187]. Diesbezüglich werden für das Loosely Coupled Workflows-Prinzip Modellierungsrahmenwerke beschrieben, welche auch den Aspekt einer serviceorientierten Architektur und Webservice-Spezifikationen berücksichtigen [15]. Ein State of the Art-Bericht diskutiert in dieser Hinsicht die Sicherstellung eines interoperablen Datenaustausches zwischen verteilt verwalteten Systemen über standardisierte Rahmenwerke wie RosettaNet[1] oder ebXML[2] [101].

dezentrale Prozessausführung

Entgegen der Auffassung dieser Arbeit wird in der Literatur zum Teil auch bei der dezentralen Prozessausführung ein zentral verwalteter Orchestrator zur Steuerung angenommen [125]. Eine weitere Umsetzungsstrategie bei der dezentralen Prozessausführung ähnelt dem Case Transfer-Workflow-Interoperabilitätsmuster, wonach mehrere verteilte WFMSs die Kontrolle stets direkt an ein weiteres WFMS abgeben [117], ohne dass ein gemeinsamer Prozessstatus verwaltet wird. Dies findet unter anderem im IoT-Kontext Anwendung, wenn die Prozessausführung aufgrund einer instabilen Konnektivität zwischen den IoT-Geräten, welche zur Erledigung von Aktivitäten angebunden sind, und einem zentral orchestrierenden WFMS beeinträchtigt werden könnte [59].

dezentrale Verwaltung eines globalen Prozessstatus

Anstatt zwischenbetriebliche Prozesse als Integration verteilter Webservices aufzufassen oder die dezentrale Prozessausführung mittels zentralen Koordinator umzusetzen, konzentriert sich diese Arbeit auf die dezentrale Ausführung eines konsolidierten Prozessmodells. Die Umsetzung erfolgt auf Basis eines dezentralen Peer-to-Peer-Netzwerks, worin alle Knoten gleichberechtigt sind und alle relevanten Informationen zur Prozessausführung in einem replizierten Datenstand verwalten, der über fehlertolerante Synchronisierungsalgorithmen koordiniert wird. Dies kann beispielsweise über Blockchain-basierte Systeme (Kapitel 5) oder über traditionellere Algorithmen (Kapitel 6) erfolgen. Dementsprechend steht statt einer nachrichtenbasierten Kollaboration die WFMS-ähnliche Ausführung eines Prozessmodells im

1 https://www.gs1us.org/resources/rosettanet, besucht am 23.02.2024
2 http://www.ebxml.org/, besucht am 23.02.2024

Fokus, insbesondere unter Berücksichtigung der wichtigsten Prozessperspektiven. Der organisatorische Aspekt der zwischenbetrieblichen Prozessausführung wird hingegen in dieser Arbeit weniger stark fokussiert, da die dezentralen Ausführungsprinzipien die kollaborative Zusammenarbeit unabhängiger Abteilungen auch innerhalb eines Unternehmens nicht ausschließt.

9.2.2 *State of the Art vor dieser Forschungsarbeit*

Loosely Coupled Workflows

Zum Zeitpunkt des Beginns dieser Forschungsarbeit wurden wie oben beschrieben zwischenbetriebliche Prozesse eher auf Basis des Loosely Coupled Workflows-Prinzip umgesetzt, während der Begriff der dezentralen Prozessausführung weniger im zwischenbetrieblichen Kontext verwendet wird und stattdessen einen zentralen Koordinator involviert. Dabei fehlt es den nachrichtenbasierten Kollaborationen an Möglichkeiten zur Nachvollziehbarkeit des Fortschritts der Prozessausführung oder weiteren grundlegenden Funktionalitäten wie kollaborationsübergreifenden personalisierten Arbeitslisten.

Blockchain-basierte Lösungen

Die alternative Blockchain-basierte Prozessausführung erlaubt es hingegen, zwischenbetriebliche Prozesse auf Basis einer dezentralen Verwaltung eines gemeinsamen Datenbestandes umzusetzen. Innerhalb der dezentralen Blockchain-basierten Prozessausführung existieren aus BPM-Sicht wiederum verschiedene Umsetzungsmöglichkeiten. Hinsichtlich der Modellierung ist die nachrichtenbasierte von der prozessbasierten Koordination zu unterscheiden. Aus Implementierungssicht existieren globale Konsensverfahren wie Proof of Work, welche durch monetäre Anreize getrieben sind, oder stärker zentralisierte Konsensusverfahren wie Proof of Authority, welche ein gewisses Vertrauen in die Blockersteller voraussetzen. Die Entwicklung der Blockchain-basierten Prozessausführung war noch nicht weit fortgeschritten, sodass die Umsetzung über einen kompilierenden Ansatz erfolgte, welcher Schwächen unter anderem im Bereich der Skalierbarkeit oder der flexiblen Integration weiterführender Prozessperspektiven aufzeigt. Darüber hinaus ist damit die Reimplementierung von WFMS-Funktionalitäten für jede neue Modellierungssprache und für jedes neue Blockchain-Protokoll erforderlich.

flexible dezentrale Prozesskontrolldatenverwaltung

Die lokale Interpretation eines Prozessmodells in einem WFMS, welches über fehlertolerante Algorithmen mit anderen WFMSs synchronisiert wird, wurde bis dato noch nicht betrachtet. Dieses Vorgehen erlaubt eine zwischenbetriebliche Prozessausführung, wobei insbesondere die Berücksichtigung wichtiger Prozessperspektiven einen essenziellen Beitrag dieser Arbeit darstellt. Es werden sowohl ein Prototyp einer rein Blockchain-basierten Lösung vorgestellt, bei dem die lokale Interpretation im Rahmen der Validierung von Transaktionen erfolgt als auch das dpex-Framework, welches eine fortgeschrittene Möglichkeit mit flexiblen Konfigurationsoptionen bietet und die Interpretati-

on in existierenden WFMSs erlaubt, während die Blockchain lediglich als Komponente zur Synchronisierung von Ereignissen angebunden wird.

9.2.3 *Beitrag dieser Forschungsarbeit*

Die Hauptbeiträge der Forschungsarbeit im Rahmen dieser Dissertation ergeben sich aus den Publikationen [166–170] und werden in der folgenden Auflistung kurz wiedergegeben.

- Zuerst wird der interpretierende Ansatz zur Blockchain-basierten Prozessausführung von BPMN-Prozessdiagrammen auf Basis der Ethereum-Blockchain vorgeschlagen. Zur Demonstration der Umsetzbarkeit wird außerdem ein Smart Contract als Artefakt eines DSR-Zyklus bereitgestellt [170].
- Es wird ferner gezeigt, wie sich die organisationale Perspektive [168] sowie datenbasierte Entscheidungen [169] an exklusiven Gattern in dem vorgestellten Smart Contract rudimentär umsetzen lassen.
- Auf Basis der Blockchain-basierten Prozessausführung wird mit Decentralized Control eine neue Form von Workflow-Interoperabilität beschrieben. Dabei erfolgt die Ausführung eines zwischenbetrieblichen Prozessmodells über die jeweils lokale Interpretation in einem WFMSs, welche über fehlertolerante Algorithmen synchronisiert werden, um dadurch den Prozessstatus dezentral zu verwalten [169].
- Eine Analyse eruiert, dass die Anforderungen der BPM-Domäne den Einsatz von globalen Blockchain-Protokollen nicht uneingeschränkt rechtfertigen und alternative Lösungen mit traditionelleren fehlertoleranten Synchronisierungsalgorithmen und digitalen Signaturen potenziell besser geeignet sind [166].
- Aufgrund der Analyse und weiteren identifizierten Schwächen von Blockchain-basierten Lösungen wird das dpex-Framework als abstraktes Konzept vorgestellt und zur Demonstration der Umsetzbarkeit wiederum durch eine Implementierung vervollständigt [167]. Mit Beispieladapterimplementierungen wird gezeigt, wie auf Basis des dpex-Frameworks eine dezentrale Prozessausführung umgesetzt werden kann, indem ein globales zwischenbetriebliches Prozessmodell in jeweils lokal laufenden Camunda WFMSs interpretiert wird, welche durch die Ethereum-Blockchain als SCI synchronisiert werden. Insbesondere werden über die Adapter Erweiterungsmöglichkeiten der externen Komponenten demonstriert, nämlich bezüglich einer fortgeschrittenen Betrachtung der organisationalen Perspektive für das Camunda WFMS in Zusammenhang mit BPMN und bezüglich der

Berücksichtigung der probabilistischen Finalität bestimmter Konsensverfahren für die Ethereum-Blockchain.

Diese aufgelisteten Hauptbeiträge werden in dieser Ausarbeitung noch um die nachstehend beschriebenen Punkte ergänzt.

- Es erfolgt eine Aufbereitung der wichtigsten Prozessperspektiven, während die Umsetzbarkeit auf Basis von BPMN-Prozessdiagrammen und dem Camunda WFMS exemplarisch gezeigt wird.
- Es werden verschiedene Möglichkeiten zur zwischenbetrieblichen Prozessausführung dargestellt. Dabei werden mit den Kombinationen aus zentral verwalteten, verteilten und dezentral verwalteten IT-Infrastrukturen sowie den nachrichtenbasierten und prozessbasierten Modellierungsparadigmen verschiedene Umsetzungsmöglichkeiten unter Berücksichtigung des Sicherheitsaspekts diskutiert.
- Eine systematische Literatur-Übersichtsarbeit zeigt den aktuellen Forschungsstand im Bereich der Blockchain-basierten Prozessausführung. Dabei erfolgt eine Kategorisierung der selektierten Veröffentlichungen hinsichtlich bestimmter Kriterien, etwa die verwendete Modellierungssprache, unterstützte Blockchain-Protokolle oder Flexibilitätskriterien. In diesem Zuge werden weitere Forschungsarbeiten im interdisziplinären Schnittpunkt der BPM- und Blockchain-Domäne vorgestellt, zum Beispiel im Bereich der Modellierung, Überwachung oder Analyse von Prozessen. Außerdem werden weitere systematische Literatur-Übersichtsarbeiten zusammengefasst.

9.3 LIMITATIONEN UND ANKNÜPFPUNKTE FÜR KÜNFTIGE FORSCHUNGSARBEITEN

Die vorgeschlagenen Konzepte und Beiträge erfahren breite Akzeptanz in der Forschungsgemeinschaft, was sich einerseits durch die konstanten Zitierungen der Veröffentlichungen zeigt[3] und andererseits durch die Weiterentwicklung [114] und Integration der Ansätze in bereits bestehende Arbeiten [109]. Weiterhin zeigt sich eine aktive Weiterentwicklung des dpex-Frameworks als Open-Source-Projekt in den öffentlichen GitLab-Repositorien, unter anderem im Rahmen von Abschlussarbeiten [41, 58, 81, 191].

In den zwei DSR-Zyklen sind zwei Artefakte beziehungsweise Prototypen entstanden. DSR-Projekte sind jedoch per definitionem zyklisch, sodass in der Evaluationsphase stets neue Herausforderungen identifiziert werden können und dadurch Anknüpfpunkte für

3 https://scholar.google.de/citations?user=dgTinnUAAAAJ&hl=de, besucht am 05.03.2024

zukünftige Arbeiten entstehen. Dieses Kapitel diskutiert die Limitationen der Beiträge dieser Arbeit, um mögliche Einstiegspunkte für weitere Forschungsarbeiten aufzuzeigen. Dies erfolgt strukturiert nach den jeweiligen Teildisziplinen der Blockchain-basierten Prozessausführung, dem dpex-Framework, der Analyse verwandter Arbeiten und der konzeptionellen Aufarbeitung der zwischenbetrieblichen Prozessausführung.

Blockchain-basierte Prozessausführung

Die Blockchain-basierte Prozessausführung bietet eine Möglichkeit, Geschäftsprozesse direkt auf Basis einer Blockchain vertrauenswürdig auszuführen. Der interpretierende Ansatz aus Kapitel 5 oder das Caterpillar-System [113] stehen exemplarisch für verschiedene Umsetzungsmöglichkeiten. Es besteht allerdings noch immer eine Diskrepanz zwischen dem Funktionsumfang etablierter WFMSs und den Prozessausführungsfunktionalitäten der Blockchain-basierten Ansätze. Dies wird unter anderem in den folgenden offenen Herausforderungen im Bereich der Blockchain-basierten Prozessausführung diskutiert.

BPMN-KONFORMITÄT Die Prozessausführung mittels des in [170] entwickelten Smart Contracts entspricht nicht zu 100% der Semantik des BPMN-Standards [113]. So dürfen nach der Evaluation eines inklusiven Oder-Gatters nur Aktivitäten ausgeführt werden, welche die auf den ausgehenden Sequenzflüssen annotierten Bedingungen erfüllen. In der Implementierung, verfügbar auf GitHub[4], wird lediglich beim entsprechenden zusammenführenden Oder-Gatter geprüft, ob mehr als ein Pfad bereits abgeschlossen ist, was auch die Ausführung alle Pfade erlaubt. In der aktuellen Version, welche in dieser Arbeit in Kapitel 5 referenziert wird, wird die Unterstützung des inklusiven Oder-Gatters daher ausgeschlossen.

Als Lösung können in zukünftigen Versionen die Bedingungen an den Sequenzflusspfeilen in das Requirementsmodell integriert werden, um die korrekte Interpretation des inklusiven Oder-Gatters von BPMN zu ermöglichen. Allerdings erlaubt inzwischen auch das Blockchain-basierte WFMS Caterpillar eine interpretierte Ausführung über das ausgereifte Token-Spiel, sodass eine Weiterentwicklung des Requirementsmodells zur Implementierung der BPMN-Semantik zuvor geprüft werden sollte.

BPMN-ELEMENTE Da bei rein Blockchain-basierten Ansätzen die Prozessausführung und Semantik von Modellierungssprachen von

4 https://github.com/Jonasmpi/PExSCo/blob/master/contracts/ContractCollaborationManager.sol, besucht am 23.02.2024

Grund auf neu implementiert werden muss, unterstützen die aktuellen Ansätze nur ausgewählte Elemente des BPMN-Standards, wie oben anhand des inklusiven Oder-Gatters gezeigt, sodass in künftigen Arbeiten die Implementierung weiterer Konstrukte manuell erfolgen muss. Die alternative Implementierung der BPMN-Semantik nach dem vorgeschlagenen Token-Spiel ermöglicht eine ausgereiftere Implementierung der BPMN-Semantik, was eine unkompliziertere Unterstützung weiterer BPMN-Elemente erlaubt, wie ebenfalls am WFMS Caterpillar zu sehen ist.

PROZESSPERSPEKTIVEN Die wichtigsten Prozessperspektiven sind in den aktuellen Ansätzen der Blockchain-basierten Prozessausführung bislang nicht vollständig unterstützt. Zum Teil ist in der organisationalen Perspektive eine rollenbasierte Allokation von menschlichen Agenten zu Aktivitäten zwar möglich, jedoch wird oftmals kein flexibles Organisationsmodell unterstützt. Andererseits zeigt die abstimmungsbasierte Allokation von Ressourcen eine neuartige kollaborative Zuweisungsmöglichkeit. Die Integration weiterer Perspektiven wie der operationalen Perspektive erfordert durch die Abgeschlossenheit der Blockchain weiterführende Konzepte wie etwa Oracles, welche in den meisten hier analysierten Ansätzen nicht berücksichtigt sind.

Es zeigt sich, dass durch die Notwendigkeit der Neuimplementierung der Prozessausführung manche Prozessperspektiven bis jetzt nicht unterstützt sind und dass durch die Blockchain-spezifischen Charakteristika spezielle Konzepte zu integrieren sind, welche in künftigen Arbeiten genauer zu entwickeln und analysieren sind.

BLOCKCHAIN-FORKS Ein weiterer Punkt betrifft die probabilistische Finalität von bestimmten Konsensverfahren in Blockchain-Protokollen, wenn Transaktionen durch die Auflösung von entstandenen Blockchain-Forks revidiert und aus der Blockchain gelöscht werden. Zwar existieren Ansätze, welche diese Herausforderung teilweise aufgreifen [45, 46], jedoch sind praxisnahe Fragen weiterhin nicht untersucht. Beispielsweise ist bisher nicht diskutiert, wie mit nicht wiederholbaren manuellen Aktivitäten, etwa dem Zerstören von Bauteilen, umgegangen werden kann, wenn die entsprechende Transaktion revidiert und der Prozessstatus somit zurückgesetzt wird. Ein anderes Beispiel betrifft die Vermeidung der wiederholten Ausführung automatisierter Aufgaben.

Zusammengefasst unterstützen einerseits die Blockchain-basierten Lösungen bisher nur einen Bruchteil der Funktionalitäten von ausgereiften traditionellen WFMSs. Andererseits sind in künftigen Arbeiten noch sowohl neuartige Möglichkeiten für die Prozessausführung wie

die kollaborative Abstimmung für Verantwortlichkeiten zu evaluieren als auch die Auswirkungen der Blockchain-Charakteristika auf die Prozessausführung zu analysieren.

dpex-Framework

Das dpex-Framework erlaubt die flexible Integration von bestehenden WFMSs und SCIs, zum Beispiel Blockchains, zur dezentralen Prozessausführung. Einerseits vermeidet dpex dadurch die Notwendigkeit der Neuimplementierung von WFMS-Funktionalitäten im Rahmen der Blockchain-basierten Prozessausführung und bietet andererseits auch die Möglichkeit einer unkomplizierten Integration von traditionelleren fehlertoleranten Synchronisierungsalgorithmen in die dezentrale Prozessausführung. Durch den Middleware-basierten Ansatz können mittels Adapterimplementierungen leicht beliebige externe Systeme, welche die geforderten Schnittstellen bedienen können, angebunden und darüber hinaus auch erweitert werden. Dies ermöglicht im Gegensatz zur rein Blockchain-basierten Prozessausführung die Integration fortgeschrittener Konzepte, unter anderem die verschiedenen Prozessperspektiven, in die dezentrale Prozessausführung.

Allerdings bietet auch das dpex-Framework Potenzial für künftige Weiterentwicklungen und Verbesserungen, wie die folgende Auflistung zeigt.

1:M-KARDINALITÄT ZWISCHEN ALLIANCE UND PROCESSMODEL In der aktuellen Version des dpex-Frameworks besteht eine 1:1-Beziehung zwischen einem kollaborationsbeschreibenden `Alliance`-Objekt und einem `ProcessModel`-Objekt, sodass für jeden Prozess einer Kollaboration ein neues `Alliance`-Objekt erstellt werden muss. In künftigen Versionen kann die Kardinalität dahingehend angepasst werden, um mehrere Prozessmodelle innerhalb einer Kollaboration zu verwalten.

KOMMUNIKATION ÜBER MEHRERE BLOCKCHAINS Manche Konzepte in verwandten Arbeiten erlauben innerhalb einer Kollaboration eine Kommunikation über mehrere Blockchain-Netzwerke hinweg (Kapitel 8). Dies betrifft teilweise die Blockchain-basierte Umsetzung des Loosely Coupled Workflow-Prinzips mit der Blockchain als Koordinator, weswegen eine sinnvolle Anwendbarkeit im Rahmen der dezentralen Prozessausführung nach Interpretation dieser Arbeit zuerst evaluiert werden muss. Gegebenenfalls kann das dpex-Framework dann dahingehend angepasst werden, dass sich eine SCI auch über mehrere Blockchain-Netzwerke erstrecken kann.

EVALUATION DES FRAMEWORKS ÜBER FALLSTUDIEN Die abstrakte Idee des dpex-Frameworks der Middleware-getriebenen Schich-

tentrennung wird im Rahmen eines DSR-Zyklus über die Demonstration der Umsetzbarkeit mittels eines implementierten Prototyps evaluiert. Die Umsetzbarkeit eines Konzepts stellt einen essenziellen Beitrag dar, denn darüber kann in künftigen Forschungsarbeiten eine Evaluation der Anwendbarkeit des Frameworks in verschiedenen Szenarien und Anwendungsfällen erfolgen. Derartige Fallstudien können zur Akzeptanz und dem Verständnis der Funktionsweise und Anwendbarkeit des Frameworks beitragen und sind ein interessanter Anknüpfpunkt für weitere Forschungsarbeiten.

EVALUATION DER IMPLEMENTIERUNG Zur Evaluation der Umsetzbarkeit von dpex erfolgt im DSR-Demonstrationsschritt eine Implementierung des Frameworks. In diesem DSR-Zyklus wird die Implementierung per se allerdings nicht evaluiert. Weitere Evaluationsschritte können unter anderem über Nutzerstudien im Bereich der Behavioural Science erfolgen [70]. Außerdem können auch quantitative Indikatoren in Bezug auf Laufzeitmessungen oder das Verhalten unter Last evaluiert werden, um die Implementierung des dpex-Frameworks besser bewerten zu können und für einen produktiven Einsatz vorzubereiten.

Implementierung weiterer Adapter

ERWEITERUNG DER IMPLEMENTIERUNG Das dpex-Framework überzeugt durch die Möglichkeit der flexiblen Anbindung von existierenden BPM- und SCI-Komponenten. In dieser Arbeit werden hierfür vorrangig das WFMS Camunda und die Ethereum-Blockchain betrachtet. In zukünftigen Arbeiten kann gezeigt werden, dass weitere WFMSs und Blockchains beziehungsweise alternative fehlertolerante Synchronisierungsalgorithmen an dpex angebunden werden können. In aktuell laufenden oder bereits abgeschlossenen Abschlussarbeiten ist gezeigt, dass unter anderem das WFMS *jBPM* als BPM-Komponente [58], Hyperledger als weitere Blockchain sowie BFT-SMaRt als nicht Blockchain-basierte SCI in dpex integriert werden können [58, 81].

Benutzerverwaltung

In der aktuellen Version integriert die `dpex-demo`-Anwendung kein fortgeschrittenes Konzept zur Benutzerverwaltung. Insbesondere wird die Signierung von Transaktionen nicht über den privaten Schlüssel des Benutzers, sondern über den externen Ethereum-Knoten abgewickelt (Kapitel 6.6). In zukünftigen Arbeiten ist einerseits der Signierungsvorgang anzupassen, um die Sicherheitskomponente adäquat umzusetzen, und außerdem sind die verschiedenen Identitäten eines Benutzers in dpex, nämlich die bpmId im Prozessmodell, die sciId für die SCI und gegebenenfalls die commId für eine Kommunikationserweiterung in ein konsolidiertes Benutzerverwaltungskonzept zu integrieren. Dabei sind unter anderem auch die Fragen der Benutzerauthentifizierung zu beantworten, etwa ob für jede Benutzeraktion die

Identität manuell überprüft werden muss oder ob die verschiedenen Identitäten von der Anwendung verwaltet werden können.

Dies wird in einer aktuellen Abschlussarbeit über die Integration und Anpassung des Benutzerverwaltungssystems *Keycloak*[5] evaluiert [81].

Aktivitätslebenszyklus

UNTERSTÜTZUNG WEITERER BPM-KONZEPTE Trotz der Anbindung und Verwendung von fortgeschrittenen WFMSs unterstützt dpex nicht ohne Weiteres deren vollständigen Funktionsumfang. Zwar kann so die Interpretation von Prozessmodellen ausgelagert werden, sodass die Semantik nicht neu implementiert werden muss, allerdings limitieren etwa die über dpex versendeten Nachrichten den Funktionsumfang. Als Beispiel wird in Kapitel 6.4.4 der Lebenszyklus von Aktivitäten diskutiert. In der aktuellen Version können in den ausgetauschten Nachrichtenobjekten der dpex-Instanzen lediglich *claim* und *complete* spezifiziert werden. Wenn weitere Phasen wie *pause*, *cancel* oder *reassign* unterstützt werden sollen, müssen entweder die Nachrichtenobjekte erweitert werden oder generell die Schnittstellen flexibler gestaltet werden.

Workflow Patterns

Es existieren noch weiterführende Konzepte im Bereich der Prozessausführung, welche in dieser Arbeit nicht berücksichtigt sind. Dazu zähle das *Multi-Instance*-Muster, welches Abhängigkeiten zwischen verschiedenen Prozessinstanzen eines Prozessmodells herstellt, wodurch die abzuarbeitenden Aktivitäten beeinflusst werden können [1]. In zukünftigen Arbeiten kann evaluiert werden, welche Anpassungen für die Unterstützung weiterer Konstrukte im dpex-Framework gegebenenfalls notwendig sind.

INTEGRATION WEITERER KRYPTOGRAFISCHER KONZEPTE Das Sicherheitsmodul wird in der aktuellen Version über einfache digitale Signaturen definiert. Im Bereich der Kryptografie existieren fortgeschrittene Konzepte, welche möglicherweise ebenfalls einen Beitrag zu einer sicheren dezentralen Prozessausführung liefern können. Dazu zählen unter anderem Gruppensignaturen [22], Ringsignaturen [198], Self-Sovereign Identities [142] oder Zero-Knowledge Proofs [49], welche für die Integration und Verwendung in dpex in künftigen Arbeiten evaluiert werden können.

KOMMUNIKATION DER PROZESSAKTEURE Ein weiteres Beispiel zur Erweiterung des dpex-Frameworks ist mit der Integration einer pseudonymisierten Kommunikation zwischen Prozessakteuren im Rahmen einer Abschlussarbeit erfolgt [41]. Damit können

5 https://www.keycloak.org/, besucht am 23.02.2023

Prozessakteure über eine Chat-Funktionalität Kontakt zu verantwortlichen Person aufnehmen, zum Beispiel über die Auswahl einer bereits abgeschlossenen Aktivität, was aufgrund der Pseudonymisierung und Verteilung aller Prozessakteure über verschiedene Unternehmen hinweg bei der dezentralen Prozessausführung ansonsten nicht ohne Weiteres möglich ist. Dafür können analog zu den externen BPM- oder SCI-Komponenten Kommunikationsdienste wie *Discord*[6] oder das dezentrale *Matrix*-Protokoll[7] über eine Adapterimplementierung an dpex angebunden und für die Verwendung von Kollaborationen konfiguriert werden. Zur Umsetzung müssen im Organisationsmodell einer bpmId eine entsprechende commId zugeordnet werden, worüber ein Prozessakteur innerhalb des Kommunikationsdienstes identifiziert werden kann.

PROZESSPERSPEKTIVEN Als reine Middleware wird die Unterstützung der Prozessperspektiven im dpex-Framework grundsätzlich über den Funktionsumfang in den extern angebundenen Komponenten definiert. So kann eine Kollaboration, welche den zwischenbetrieblichen Prozess rein über ein BPMN-Prozessdiagramm spezifiziert, zunächst keine komplexen Zuweisungsregeln von Aktivitäten zu organisationalen Entitäten definieren. Allerdings können die externen Komponenten über die Adapterimplementierungen erweitert werden, wie in dieser Arbeit mit der OrgEngine am Beispiel der organisationalen Perspektive für BPMN und Camunda gezeigt wird. Über eine Evaluation von dpex mittels Fallstudien bleibt zu prüfen, inwiefern sich die hier verwendeten tabellarischen Organisationsimplementierungsmodelle in der Praxis eignen. Gegebenenfalls erlaubt dpex eine flexible Anpassung der Adapterimplementierung. Die informationsorientierte Perspektive wird in dieser Arbeit nicht im Detail diskutiert. Im Rahmen einer weiteren Abschlussarbeit ist jedoch bereits gezeigt, wie die Verwaltung von Datenwerten im dpex-Framework umgesetzt werden kann [58]. Dabei werden Prozessvariablen über die SCI sicher synchronisiert und können so im externen Camunda WFMS organisiert und verwendet werden. In künftigen Arbeiten sind fortgeschrittene Datenmanagementkonzepte wie dezentrale Dateisysteme oder Oracles für die Anbindung an dpex zu evaluieren. Wie in Kapitel 6.6 diskutiert, sind für eine vollständige Unterstützung der operationalen Perspektive im dpex-Framework gegebenenfalls Anpassungen an den externen WFMSs beziehungsweise den BPM-Konnektoren vorzunehmen. Insbesondere stellen die Anbindung lokaler oder kollaborationsübergreifender Systeme

organisationale Perspektive

informationsorientierte Perspektive

operationale Perspektive

6 https://discord.com/, besucht am 25.02.2024
7 https://matrix.org/, besucht am 25.02.2024

Herausforderungen dar, welche in künftigen Arbeiten zu lösen sind oder bereits in aktuellen Arbeiten evaluiert werden [191].

Verwandte Arbeiten

PROZESSPERSPEKTIVEN In Kapitel 8 werden einige Ansätze der Blockchain-basierten Prozessausführung diskutiert und nach bestimmten Kategorien eingeordnet. Dies erfolgt allerdings ohne Berücksichtigung des unterstützten Umfangs hinsichtlich der Prozessperspektiven. In weiteren Analysen können die unterschiedlichen Ansätze dahingehend untersucht werden, inwieweit und inwiefern die Perspektiven in die jeweiligen Architekturen oder Implementierungen integriert sind.

Zwischenbetriebliche Prozessausführung

In dieser Arbeit steht, anstatt sicherheitsrelevante und organisatorische Fragestellungen über die Anwendbarkeit zu beantworten, die technische Umsetzung der dezentralen Prozessausführung im Fokus. Dementsprechend bleiben diese Diskussionen als offene Forschungsfragen für künftige Arbeiten offen und werden im Folgenden kurz zusammengefasst.

In dieser Arbeit werden verschiedene Möglichkeiten aufgezeigt, zwischenbetriebliche Prozesse umzusetzen. Dabei lassen sich nachrichtenbasierte von prozessbasierten Modellierungskonzepten unterscheiden, welche sich jeweils auf zentral verwalteten, verteilten und dezentral verwalteten Infrastrukturen umsetzen lassen (Kapitel 3). Weiterhin existieren innerhalb dieser Kombinationen wiederum verschiedene Implementierungsmöglichkeiten. Beispielsweise kann die Umsetzung von nachrichtenbasierten Modellen auf einer verteilten Infrastruktur entweder auf Basis einer serviceorientierten Architektur mit Webservices erfolgen oder über den Nachrichtenaustausch über eine REST-Schnittstelle unter Verwendung des HTTP-Protokolls zur Koordination lokal ausgeführter Prozesse. Andererseits zeigt diese Arbeit auch die Vielfalt an Architekturen im Bereich der dezentralen Prozessausführung. Sowohl nachrichtenbasierte Choreografien als auch Prozessmodelle lassen sich darüber implementieren. Unabhängig vom Modellierungsparadigma stehen für die Umsetzung der dezentralen Prozessausführung verschiedene Systeme und Algorithmen zur Verfügung, darunter Blockchain-Protokolle oder traditionellere fehlertolerante Synchronisierungsalgorithmen. Beim Einsatz von Blockchain-Protokollen können Prozesse einerseits direkt auf der Blockchain in Smart Contracts interpretiert werden (Fat Contract), andererseits kann die Blockchain auch lediglich zur Kommunikation und Synchronisierung von Nachrichten eingesetzt werden, womit die Prozessausführung in einem Blockchain-externen WFMS zu imple-

mentieren ist (Thin Contract). Erfolgt die Umsetzung mit Fat Contracts kann sowohl der kompilierende Ansatz als auch ein interpretierender Ansatz verfolgt werden. Zuletzt besteht Wahlfreiheit im verwendeten Blockchain-Protokoll, dessen Konfiguration oder bezüglich eines alternativen fehlertoleranten Synchronisierungsalgorithmus.

Diese Auflistung zeigt die vielfältigen Umsetzungsmöglichkeiten zwischenbetrieblicher Kollaborationen beziehungsweise der dezentralen Prozessausführung. In künftigen Forschungsarbeiten ist diese Variabilität systematisch und strukturiert aufzuarbeiten. Dabei ist unter anderem ein Leitfaden für die Entscheidungsfindung von Unternehmen oder Kollaborationen zu entwickeln, welche Implementierung für welche Anwendungsfälle zu verwenden ist, welche Vor- und Nachteile, etwa in Bezug auf Sicherheitsaspekte, sich dadurch ergeben oder welche Anforderungen gegebenenfalls dafür erfüllt sein müssen. Insbesondere sind hier auch Sicherheitsfragen zu diskutieren, zum Beispiel, wann eine starke Dezentralisierung auf einem öffentlichen Blockchain-Netzwerk sinnvoll ist oder wann ein privat verwaltetes Blockchain-Netzwerk vorzuziehen ist.

EIGENE VERÖFFENTLICHUNGEN

Die folgende Auflistung zeigt alle wissenschaftlichen Veröffentlichungen mit Beteiligung des Authors, wobei die im Kontext dieser Arbeit relevanten Veröffentlichungen grau hinterlegt sind.

2017

Distributed Multi-Perspective Declare Discovery
Christian Sturm, Stefan Schönig und Claudio Di Ciccio
Proceedings of the BPM Demo Track and BPM Dissertation Award co-located with 15th International Conference on Business Process Modeling (BPM 2017), Barcelona, Spain, September 13, 2017
https://ceur-ws.org/Vol-1920/BPM_2017_paper_194.pdf

2018

The Chances of including Extrinsic Factors in Business Process Management
Christian Sturm
Proceedings of the 10th Central European Workshop on Services and their Composition, Dresden, Germany, February 8-9, 2018
https://ceur-ws.org/Vol-2072/paper4.pdf

A MapReduce Approach for Mining Multi-Perspective Declarative Process Models
Christian Sturm, Stefan Schönig und Stefan Jablonski
Proceedings of the 20th International Conference on Enterprise Information Systems, ICEIS 2018, Funchal, Madeira, Portugal, March 21-24, 2018, Volume 2
https://doi.org/10.5220/0006710305850595

Mining Expressive and Executable Resource-Aware Imperative Process Models
Cristina Cabanillas, Stefan Schönig, Christian Sturm und Jan Mendling
Enterprise, Business-Process and Information Systems Modeling - 19th International Conference, BPMDS 2018, 23rd International Conference, EMM-

SAD 2018, Held at CAiSE 2018, Tallinn, Estonia, June 11-12, 2018, Proceedings
https://doi.org/10.1007/978-3-319-91704-7_1

2019

A Lean Architecture for Blockchain Based Decentralized Process Execution
Christian Sturm, Jonas Szalanczi, Stefan Schönig und Stefan Jablonski
Business Process Management Workshops - BPM 2018 International Workshops, Sydney, NSW, Australia, September 9-14, 2018, Revised Papers
https://doi.org/10.1007/978-3-030-11641-5_29

Big Data Meets Process Science: Distributed Mining of MP-Declare Process Models
Christian Sturm und Stefan Schönig
Enterprise Information Systems - 20th International Conference, ICEIS 2018, Funchal, Madeira, Portugal, March 21-24, 2018, Revised Selected Papers
https://doi.org/10.1007/978-3-030-26169-6_19

Proceedings of the 11th Central European Workshop on Services and their Composition. Bayreuth, Germany, February 14-15, 2019
Stefan Kolb und Christian Sturm
https://ceur-ws.org/Vol-2339

Full Support for Efficiently Mining Multi-Perspective Declarative Constraints from Process Logs
Christian Sturm, Myriel Fichtner und Stefan Schönig
Selected Papers from ICEIS 2018: Advances in Enterprise Information Systems
https://doi.org/10.3390/info10010029

A Blockchain-based and resource-aware process execution engine
Christian Sturm, Jonas Szalanczi, Stefan Schönig und Stefan Jablonski
Future Generation Computer Systems Volume 100 (2019)
https://doi.org/10.1016/j.future.2019.05.006

2020

Decentralized Control: A Novel Form of Interorganizational Workflow Interoperability
Christian Sturm, Jonas Szalanczi, Stefan Jablonski und Stefan Schönig
The Practice of Enterprise Modeling - 13th IFIP Working Conference, PoEM 2020, Riga, Latvia, November 25-27, 2020, Proceedings
https://doi.org/10.1007/978-3-030-63479-7_18

Interorganizational Process Execution Beyond Ethereum: Road to a Special Purpose Ecosystem
Christian Sturm und Stefan Jablonski
Proceedings of the workshops co-organized with the 13th IFIP WG 8.1 working conference on the Practice of Enterprise Modelling (PoEM 2020), Online (originally located in Riga, Latvia), November 26, 2020
https://ceur-ws.org/Vol-2749/paper6.pdf

The RALph miner for automated discovery and verification of resource-aware process models
Cristina Cabanillas, Lars Ackermann, Stefan Schönig, Christian Sturm, Jan Mendling
Software and Systems Modeling Volume 19 (2020)
https://doi.org/10.1007/s10270-020-00820-7

2023

The Dpex-Framework: Towards Full WFMS Support for Decentralized Process Execution.
Christian Sturm und Stefan Jablonski
Business Process Management Forum - BPM 2023 Forum, Utrecht, The Netherlands, September 11-15, 2023, Proceedings
https://doi.org/10.1007/978-3-031-41623-1_2

LITERATUR

[1] W. M. P. van der Aalst, A. H. M. ter Hofstede, B. Kiepuszewski und A. P. Barros. „Workflow Patterns“. In: *Distributed and Parallel Databases* 14.1 (2003), S. 5–51. ISSN: 1573-7578. DOI: 10.1023/A:1022883727209. URL: https://doi.org/10.1023/A:1022883727209.

[2] Wil M. P. van der Aalst und Mathias Weske. „The P2P Approach to Interorganizational Workflows“. In: *Advanced Information Systems Engineering*. Hrsg. von Klaus R. Dittrich, Andreas Geppert und Moira C. Norrie. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, S. 140–156. ISBN: 978-3-540-45341-3.

[3] Wil van der Aalst. *Process Mining: Data Science in Action*. 2nd. Springer Berlin, Heidelberg, 2016. ISBN: 978-3-662-49850-7. DOI: 10.1007/978-3-662-49851-4.

[4] Amal Abid, Saoussen Cheikhrouhou und Mohamed Jmaiel. „Modelling and Executing Time-Aware Processes in Trustless Blockchain Environment“. In: *Risks and Security of Internet and Systems*. Hrsg. von Slim Kallel, Frédéric Cuppens, Nora Cuppens-Boulahia und Ahmed Hadj Kacem. Cham: Springer International Publishing, 2020, S. 325–341. ISBN: 978-3-030-41568-6.

[5] Michael Adams, Suriadi Suriadi, Akhil Kumar und Arthur H. M. ter Hofstede. „Flexible Integration of Blockchain with Business Process Automation: A Federated Architecture“. In: *Advanced Information Systems Engineering*. Hrsg. von Nicolas Herbaut und Marcello La Rosa. Cham: Springer International Publishing, 2020, S. 1–13. ISBN: 978-3-030-58135-0.

[6] Paulo Henrique Alves., Ronnie Paskin., Isabella Frajhof., Yang Ricardo Miranda., João Gabriel Jardim., Jose Jorge Brum Cardoso., Eduardo Henrique Haddad Tress., Rogério Ferreira da Cunha., Rafael Nasser. und Gustavo Robichez. „Exploring Blockchain Technology to Improve Multi-party Relationship in Business Process Management Systems“. In: *Proceedings of the 22nd International Conference on Enterprise Information Systems - Volume 2: ICEIS*. INSTICC. SciTePress, 2020, S. 817–825. ISBN: 978-989-758-423-7. DOI: 10.5220/0009565108170825.

[7] Andreas Antonopoulos. *Mastering Bitcoin*. 2. Aufl. 1005 Gravenstein Highway North, Sebastopol, CA 95472: O'Reilly Media, Inc., 2017. ISBN: 978-1-491-95438-6.

[8] Andreas Antonopoulos und Gavin Wood. *Mastering Ethereum*. 1. Aufl. 1005 Gravenstein Highway North, Sebastopol, CA 95472: O'Reilly Media, Inc., 2019. ISBN: 978-1-491-97194-9.

[9] Henning Baars und Hans-Georg Kemper. *Business Intelligence & Analytics – Grundlagen und praktische Anwendungen*. Wiesbaden: Springer Vieweg, 2021. ISBN: 978-3-8348-2344-1. DOI: 10.1007/978-3-8348-2344-1.

[10] Adam Back. *Hash cash postage implementation*. 1997. URL: http://www.hashcash.org/papers/announce.txt (besucht am 26. 11. 2023).

[11] Adam Back. „Hashcash - A Denial of Service Counter-Measure". In: (Sep. 2002).

[12] Alysson Bessani, João Sousa und Eduardo E.P. Alchieri. „State Machine Replication for the Masses with BFT-SMART". In: *2014 44th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*. 2014, S. 355–362. DOI: 10.1109/DSN.2014.43.

[13] Norbert Bodziony, Paweł Jemioło, Krzysztof Kluza und Marek R. Ogiela. „Blockchain-Based Address Alias System". In: *Journal of Theoretical and Applied Electronic Commerce Research* 16.5 (2021), S. 1280–1296. ISSN: 0718-1876. DOI: 10.3390/jtaer16050072. URL: https://www.mdpi.com/0718-1876/16/5/72.

[14] Nelson Bore, Andrew Kinai, Juliet Mutahi, David Kaguma, Fred Otieno, Sekou L. Remy und Komminist Weldemariam. „On Using Blockchain Based Workflows". In: *2019 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*. 2019, S. 112–116. DOI: 10.1109/BLOC.2019.8751446.

[15] Khoutir Bouchbout und Zaia Alimazighi. „Inter-Organizational Business Processes Modelling Framework". In: *Symposium on Advances in Databases and Information Systems*. 2011. URL: https://api.semanticscholar.org/CorpusID:12116400.

[16] Amina Brahem, Nizar Messai, Yacine Sam, Sami Bhiri, Thomas Devogele und Walid Gaaloul. „Running Transactional Business Processes with Blockchain's Smart Contracts". In: *2020 IEEE International Conference on Web Services (ICWS)*. 2020, S. 89–93. DOI: 10.1109/ICWS49710.2020.00019.

[17] Christoph H.-J. Braun, Janina Traue, Boris Lingl und Tobias Käfer. „Documenting the Execution of Semantically Modelled Inter-organisational Workflows on a Distributed Ledger". In: *2021 IEEE International Conference on Blockchain (Blockchain)*. 2021, S. 280–286. DOI: 10.1109/Blockchain53845.2021.00045.

[18] Christoph Bussler. *Organisationsverwaltung in Workflow-Management-Systemen*. Wiesbaden: Springer Fachmedien Wiesbaden, 1998. ISBN: 978-3-8244-2102-2.

[19] Vitalik Buterin, Diego Hernandez, Thor Kamphefner, Khiem Pham, Zhi Qiao, Danny Ryan, Juhyeok Sin, Ying Wang und Yan X Zhang. *Combining GHOST and Casper*. 2020. arXiv: 2003.03052 [cs.CR].

[20] Cristina Cabanillas, Lars Ackermann, Stefan Schönig, Christian Sturm und Jan Mendling. „The RALph miner for automated discovery and verification of resource-aware process models“. In: *Software and Systems Modeling* 19.6 (2020), S. 1415–1441. ISSN: 1619-1374. DOI: 10.1007/s10270-020-00820-7. URL: https://doi.org/10.1007/s10270-020-00820-7.

[21] Cristina Cabanillas, Stefan Schönig, Christian Sturm und Jan Mendling. „Mining Expressive and Executable Resource-Aware Imperative Process Models“. In: *Enterprise, Business-Process and Information Systems Modeling*. Hrsg. von Jens Gulden, Iris Reinhartz Berger, Rainer Schmidt, Sérgio Guerreiro, Wided Guédria und Palash Bera. Cham: Springer International Publishing, 2018, S. 3–18. ISBN: 978-3-319-91704-7.

[22] Jan Camenisch und Markus Stadler. „Efficient group signature schemes for large groups“. In: *Advances in Cryptology — CRYPTO '97*. Hrsg. von Burton S. Kaliski. Berlin, Heidelberg: Springer Berlin Heidelberg, 1997, S. 410–424. ISBN: 978-3-540-69528-8.

[23] Miguel Castro und Barbara Liskov. „Practical Byzantine Fault Tolerance“. In: *Proceedings of the Third Symposium on Operating Systems Design and Implementation*. OSDI '99. New Orleans, Louisiana, USA: USENIX Association, 1999, 173–186. ISBN: 188-0446391.

[24] Shuchih E. Chang und Yichian Chen. „When Blockchain Meets Supply Chain: A Systematic Literature Review on Current Development and Potential Applications“. In: *IEEE Access* 8 (2020), S. 62478–62494. DOI: 10.1109/ACCESS.2020.2983601.

[25] Michele Chinosi und Alberto Trombetta. „BPMN: An introduction to the standard“. In: *Computer Standards & Interfaces* 34.1 (2012), S. 124–134. ISSN: 0920-5489. DOI: https://doi.org/10.1016/j.csi.2011.06.002. URL: https://www.sciencedirect.com/science/article/pii/S0920548911000766.

[26] Henri Cohen. *A Course in Computational Algebraic Number Theory*. Springer-Verlag Berlin Heidelberg New York, 1993.

[27] F. Corradini, A. Marcelletti, A. Morichetta, A. Polini, B. Re und F. Tiezzi. „Engineering Trustable Choreography-Based Systems Using Blockchain“. In: *Proceedings of the 35th Annual ACM Symposium on Applied Computing*. SAC '20. Brno, Czech Republic: Association for Computing Machinery, 2020, 1470–1479. ISBN:

9781450368667. DOI: 10.1145/3341105.3373988. URL: https://doi.org/10.1145/3341105.3373988.

[28] Flavio Corradini, Alessandro Marcelletti, Andrea Morichetta, Andrea Polini, Barbara Re, Emanuele Scala und Francesco Tiezzi. „Model-driven engineering for multi-party business processes on multiple blockchains“. In: *Blockchain: Research and Applications* 2.3 (2021), S. 100018. ISSN: 2096-7209. DOI: https://doi.org/10.1016/j.bcra.2021.100018. URL: https://www.sciencedirect.com/science/article/pii/S2096720921000130.

[29] Flavio Corradini, Alessandro Marcelletti, Andrea Morichetta, Andrea Polini, Barbara Re und Francesco Tiezzi. „Engineering Trustable and Auditable Choreography-Based Systems Using Blockchain“. In: *ACM Trans. Manage. Inf. Syst.* 13.3 (2022). ISSN: 2158-656X. DOI: 10.1145/3505225. URL: https://doi.org/10.1145/3505225.

[30] Simon Curty, Felix Härer und Hans-Georg Fill. „Design of blockchain-based applications using model-driven engineering and low-code/no-code platforms: a structured literature review“. In: *Software and Systems Modeling* (2023). ISSN: 1619-1374. DOI: 10.1007/s10270-023-01109-1. URL: https://doi.org/10.1007/s10270-023-01109-1.

[31] C. Dechsupa, W. Vatanawood und A. Thongtak. „Transformation of the BPMN Design Model into a Colored Petri Net Using the Partitioning Approach“. In: *IEEE Access* 6 (2018), S. 38421–38436. DOI: 10.1109/ACCESS.2018.2853669.

[32] Gero Decker und Alistair Barros. „Interaction Modeling Using BPMN“. In: *Business Process Management Workshops*. Hrsg. von Arthur ter Hofstede, Boualem Benatallah und Hye-Young Paik. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, S. 208–219. ISBN: 978-3-540-78238-4.

[33] Claudio Di Ciccio, Alessio Cecconi, Jan Mendling, Dominik Felix, Dominik Haas, Daniel Lilek, Florian Riel, Andreas Rumpl und Philipp Uhlig. „Blockchain-Based Traceability of Inter-organisational Business Processes“. In: *Business Modeling and Software Design*. Hrsg. von Boris Shishkov. Cham: Springer International Publishing, 2018, S. 56–68. ISBN: 978-3-319-94214-8.

[34] Claudio Di Ciccio, Giovanni Meroni und Pierluigi Plebani. „On the adoption of blockchain for business process monitoring“. In: *Software and Systems Modeling* 21.3 (2022), S. 915–937. ISSN: 1619-1374. DOI: 10.1007/s10270-021-00959-x. URL: https://doi.org/10.1007/s10270-021-00959-x.

[35] L. G. Dickson. „On the Factorization of Large Numbers“. In: *The American Mathematical Monthly* 15.12 (1908), S. 217–222. DOI: 10.1080/00029890.1908.11997462.

[36] W. Diffie und M. Hellman. „New directions in cryptography“. In: *IEEE Transactions on Information Theory* 22.6 (1976), S. 644–654. DOI: 10.1109/TIT.1976.1055638.

[37] Remco M. Dijkman, Marlon Dumas und Chun Ouyang. „Semantics and analysis of business process models in BPMN“. In: *Information and Software Technology* 50.12 (2008), S. 1281–1294. ISSN: 0950-5849. DOI: https://doi.org/10.1016/j.infsof.2008.02.006. URL: https://www.sciencedirect.com/science/article/pii/S0950584908000323.

[38] Pierre Dillenbourg. *What do you mean by collaborative learning?* 1999.

[39] Marlon Dumas, Marcello La Rosa, Jan Mendling und Hajo A. Reijers. *Fundamentals of Business Process Management*. Heidelberg: Springer Berlin, 2018. ISBN: 978-3-662-56509-4. DOI: 10.1007/978-3-662-56509-4.

[40] Cynthia Dwork und Moni Naor. „Pricing via Processing or Combatting Junk Mail“. In: *Advances in Cryptology — CRYPTO' 92*. Hrsg. von Ernest F. Brickell. Berlin, Heidelberg: Springer Berlin Heidelberg, 1993, S. 139–147.

[41] Stefan am Ende. „Integration sicherer Kommunikationsschemata in das DPEX-Framework zur dezentralen Prozessausführung mithilfe des Matrix-Protokolls“. Masterarbeit (unveröffentlicht). Universität Bayreuth, 30. Jan. 2024.

[42] Johann Engelhard und Jörn Altmann. *Gabler Wirtschaftslexikon: Joint Venture*. 2018. URL: https://wirtschaftslexikon.gabler.de/definition/joint-venture-37135/version-260578 (besucht am 24. 11. 2023).

[43] Joerg Evermann. „Adapting Workflow Management Systems to BFT Blockchains - The YAWL Example“. In: *CoRR* abs/2006.05808 (2020). arXiv: 2006.05808. URL: https://arxiv.org/abs/2006.05808.

[44] Joerg Evermann und Henry M. Kim. „Workflow Management on BFT Blockchains“. In: *Enterprise Modelling and Information Systems Architectures* 15 (2020), 14:1–14:22. URL: https://api.semanticscholar.org/CorpusID:170078577.

[45] Ghareeb Falazi, Michael Hahn, Uwe Breitenbücher und Frank Leymann. „Modeling and execution of blockchain-aware business processes“. In: *SICS Software-Intensive Cyber-Physical Systems* 34.2 (2019), S. 105–116. ISSN: 2524-8529. DOI: 10.1007/s00450-019-00399-5. URL: https://doi.org/10.1007/s00450-019-00399-5.

[46] Ghareeb Falazi, Michael Hahn, Uwe Breitenbücher, Frank Leymann und Vladimir Yussupov. „Process-Based Composition of Permissioned and Permissionless Blockchain Smart Contracts“. In: *2019 IEEE 23rd International Enterprise Distributed Object Computing Conference (EDOC)*. 2019, S. 77–87. DOI: 10.1109/EDOC.2019.00019.

[47] Weidong Fang, Wei Chen, Wuxiong Zhang, Jun Pei, Weiwei Gao und Guohui Wang. „Digital signature scheme for information non-repudiation in blockchain: a state of the art review“. In: *EURASIP Journal on Wireless Communications and Networking* 2020.1 (2020), S. 56. ISSN: 1687-1499. DOI: 10.1186/s13638-020-01665-w. URL: https://doi.org/10.1186/s13638-020-01665-w.

[48] Steven Farrugia, Joshua Ellul und George Azzopardi. „Detection of illicit accounts over the Ethereum blockchain“. In: *Expert Systems with Applications* 150 (2020), S. 113318. ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2020.113318. URL: https://www.sciencedirect.com/science/article/pii/S0957417420301433.

[49] U. Fiege, A. Fiat und A. Shamir. „Zero knowledge proofs of identity“. In: *Proceedings of the Nineteenth Annual ACM Symposium on Theory of Computing*. STOC '87. New York, New York, USA: Association for Computing Machinery, 1987, 210–217. ISBN: 0897912217. DOI: 10.1145/28395.28419. URL: https://doi.org/10.1145/28395.28419.

[50] Gilbert Fridgen, Sven Radszuwill, Nils Urbach und Lena Utz. „Cross-Organizational Workflow Management Using Blockchain Technology - Towards Applicability, Auditability, and Automation“. In: *Hawaii International Conference on System Sciences*. 2018. URL: https://api.semanticscholar.org/CorpusID:27194574.

[51] *Gabler Wirtschaftslexikon: Ad-hoc-Kooperation*. 2018. URL: https://wirtschaftslexikon.gabler.de/definition/ad-hoc-kooperation-30358/version-253942 (besucht am 24. 11. 2023).

[52] *Gabler Wirtschaftslexikon: strategische Allianz*. 2018. URL: https://wirtschaftslexikon.gabler.de/definition/strategische-allianz-45084/version-268384 (besucht am 24. 11. 2023).

[53] Luciano García-Bañuelos, Alexander Ponomarev, Marlon Dumas und Ingo Weber. „Optimized Execution of Business Processes on Blockchain“. In: *Business Process Management*. Hrsg. von Josep Carmona, Gregor Engels und Akhil Kumar. Cham: Springer International Publishing, 2017, S. 130–146. ISBN: 978-3-319-65000-5.

[54] Julian Alberto Garcia-Garcia, Nicolás Sánchez-Gómez, David Lizcano, M. J. Escalona und Tomás Wojdyński. „Using Blockchain to Improve Collaborative Business Process Management: Systematic Literature Review“. In: *IEEE Access* 8 (2020), S. 142312–142336. DOI: 10.1109/ACCESS.2020.3013911.

[55] Gerhard Schewe. *Ablauforganisation*. https://wirtschaftslexikon.gabler.de/definition/ablauforganisation-27126/version-250789. Accessed: 2023-11-14. 2018.

[56] Gerhard Schewe. *Aufbauorganisation*. https://wirtschaftslexikon.gabler.de/definition/aufbauorganisation-31264/version-254826. Accessed: 2023-11-14. 2018.

[57] Paul Grefen und Remmert Remmerts de Vries. „A reference architecture for workflow management systems“. In: *Data & Knowledge Engineering* 27.1 (1998), S. 31–57. ISSN: 0169-023X. DOI: https://doi.org/10.1016/S0169-023X(97)00057-8. URL: https://www.sciencedirect.com/science/article/pii/S0169023X97000578.

[58] Tobias Gretsch. „Full WFMS- and SCI- Support for Decentralized Process Execution in the dpex-Framework“. Masterarbeit (unveröffentlicht). Universität Bayreuth, 14. Nov. 2023.

[59] Kai Grunert, Janis Joderi Shoferi, Kai Rohwer, Elitsa Pankovska und Lucas Gold. „Architecture of decentralized Process Management Systems“. In: *Business Process Management*. Hrsg. von Claudio Di Ciccio, Remco Dijkman, Adela del Río Ortega und Stefanie Rinderle-Ma. Cham: Springer International Publishing, 2022, S. 436–452. ISBN: 978-3-031-16103-2.

[60] Diksha Gupta, Jared Saia und Maxwell Young. „Proof of Work Without All the Work“. In: *Proceedings of the 19th International Conference on Distributed Computing and Networking*. ICDCN '18. Varanasi, India: Association for Computing Machinery, 2018. ISBN: 9781450363723. DOI: 10.1145/3154273.3154333. URL: https://doi.org/10.1145/3154273.3154333.

[61] Christian Gutschier, Ralph Hoch, Hermann Kaindl und Roman Popp. „A pitfall with BPMN execution“. In: *Second International Conference on Building and Exploring Web Based Environments*. 2014, S. 7–13.

[62] Stephan Haarmann. „Estimating the Duration of Blockchain-Based Business Processes Using Simulation“. In: *Central-European Workshop on Services and their Composition*. 2019. URL: https://api.semanticscholar.org/CorpusID:96426650.

[63] Stephan Haarmann, Kimon Batoulis, Adriatik Nikaj und Mathias Weske. „DMN Decision Execution on the Ethereum Blockchain“. In: *Advanced Information Systems Engineering*. Hrsg. von

John Krogstie und Hajo A. Reijers. Cham: Springer International Publishing, 2018, S. 327–341. ISBN: 978-3-319-91563-0.

[64] Stephan Haarmann, Kimon Batoulis, Adriatik Nikaj und Mathias Weske. „Executing Collaborative Decisions Confidentially on Blockchains“. In: *Business Process Management: Blockchain and Central and Eastern Europe Forum*. Hrsg. von Claudio Di Ciccio, Renata Gabryelczyk, Luciano García-Bañuelos, Tomislav Hernaus, Rick Hull, Mojca Indihar Štemberger, Andrea Kő und Mark Staples. Cham: Springer International Publishing, 2019, S. 119–135. ISBN: 978-3-030-30429-4.

[65] Abdelatif Hafid, Abdelhakim Senhaji Hafid und Mustapha Samih. „Scaling Blockchains: A Comprehensive Survey“. In: *IEEE Access* 8 (2020), S. 125244–125262. DOI: 10.1109/ACCESS.2020.3007251.

[66] Tiphaine Henry. „Towards trustworthy, flexible, and privacy-preserving peer-to-peer business process management systems. (Vers une gestion de processus métiers pair à pair fiable, flexible, et respectueuse de la vie privée)“. Diss. Polytechnic Institute of Paris, Palaiseau, France, 2022. URL: https://tel.archives-ouvertes.fr/tel-03931346.

[67] Tiphaine Henry, Roman Beck, Nassim Laga, Walid Gaaloul und Shenle Pan. „Decentralized procurement mechanisms for efficient logistics services mapping - a design science research approach“. In: *55th Hawaii International Conference on System Sciences, HICSS 2022, Virtual Event / Maui, Hawaii, USA, January 4-7, 2022*. ScholarSpace, 2022, S. 1–10. URL: http://hdl.handle.net/10125/79952.

[68] Tiphaine Henry, Nassim Laga, Julien Hatin, Roman Beck und Walid Gaaloul. „Hire me fairly: towards dynamic resource-binding with smart contracts“. In: *2021 IEEE International Conference on Services Computing (SCC)*. 2021, S. 407–412. DOI: 10.1109/SCC53864.2021.00058.

[69] Tiphaine Henry, Nassim Laga, Julien Hatin, Walid Gaaloul und Imed Boughzala. „Cross-Collaboration Processes based on Blockchain and IoT: a survey“. In: *54th Hawaii International Conference on System Sciences, HICSS 2021, Kauai, Hawaii, USA, January 5, 2021*. ScholarSpace, 2021, S. 1–10. URL: https://hdl.handle.net/10125/71138.

[70] Alan R. Hevner, Salvatore T. March, Jinsoo Park und Sudha Ram. „Design Science in Information Systems Research“. In: *MIS Quarterly* 28.1 (2004), S. 75–105. ISSN: 02767783. URL: http://www.jstor.org/stable/25148625 (besucht am 23. 08. 2023).

[71] Thomas T. Hildebrandt und Raghava Rao Mukkamala. „Declarative Event-Based Workflow as Distributed Dynamic Condition Response Graphs“. In: *Proceedings Third Workshop on Programming Language Approaches to Concurrency and communication-cEntric Software, PLACES 2010, Paphos, Cyprus, 21st March 2010*. Hrsg. von Kohei Honda und Alan Mycroft. Bd. 69. EPTCS. 2010, S. 59–73. DOI: 10.4204/EPTCS.69.5. URL: https://doi.org/10.4204/EPTCS.69.5.

[72] David Hollingsworth. *The Workflow Reference Model*. The Workflow Management Coalition, Jan. 1995. URL: http://www.workflow\-patterns.com/documentation/documents/tc003v11.pdf.

[73] Richard Hull, Vishal S. Batra, Yi-Min Chen, Alin Deutsch, Fenno F. Terry Heath III und Victor Vianu. „Towards a Shared Ledger Business Collaboration Language Based on Data-Aware Processes“. In: *Service-Oriented Computing*. Hrsg. von Quan Z. Sheng, Eleni Stroulia, Samir Tata und Sami Bhiri. Cham: Springer International Publishing, 2016, S. 18–36. ISBN: 978-3-319-46295-0.

[74] Florian Idelberger, Guido Governatori, Régis Riveret und Giovanni Sartor. „Evaluation of Logic-Based Smart Contracts for Blockchain Systems“. In: *Rule Technologies. Research, Tools, and Applications*. Hrsg. von Jose Julio Alferes, Leopoldo Bertossi, Guido Governatori, Paul Fodor und Dumitru Roman. Cham: Springer International Publishing, 2016, S. 167–183. ISBN: 978-3-319-42019-6.

[75] Md. Mainul Islam, Mpyana Mwamba Merlec und Hoh Peter In. „A Comparative Analysis of Proof-of-Authority Consensus Algorithms: Aura vs Clique“. In: *2022 IEEE International Conference on Services Computing (SCC)*. 2022, S. 327–332. DOI: 10.1109/SCC55611.2022.00054.

[76] Stefan Jablonski. „MOBILE: A modular workflow model and architecture“. In: *Proceedings of the Fourth International Working Conference on Dynamic Modelling and Information Systems*. Hrsg. von Alexander Verbraeck, Henk G. Sol und Pieter W.G. Bots. Delft University Press, 1994. ISBN: 90-407-1051-1. URL: https://www.researchgate.net/publication/2720558_MOBILE_A_modular_workflow_model_and_architecture.

[77] Stefan Jablonski. *Workflow-Management-Systeme. Modellierung und Architektur*. Bonn: International Thomson Publishing GmbH, 1995. ISBN: 3-8266-0129-7.

[78] Stefan Jablonski und Christoph Bussler. *Workflow Management: Modeling Concepts, Architecture and Implementation*. London: International Thomson Computer Press, 1996. ISBN: 978-1-85032-222-1.

[79] Markus Jakobsson und Ari Juels. „Proofs of Work and Bread Pudding Protocols“. In: *Proceedings of the IFIP TC6/TC11 Joint Working Conference on Secure Information Networks: Communications and Multimedia Security*. CMS '99. NLD: Kluwer, B.V., 1999, 258–272.

[80] Barkha Javed. „A generic provenance framework to document public policy making processes“. Diss. URL: https://uwe-repository.worktribe.com/output/4763638.

[81] Moritz Johlige. „dpex Framework: Advancing from Proof-of-Concept to Production Readiness“. Masterarbeit (in Bearbeitung). Universität Bayreuth.

[82] Haruhiko Kaiya, Tomoya Misawa, Shinpei Ogata, Shinobu Saito, Hiroyuki Nakagawa und Hironori Takeuchi. „A Proposal to Find Mutually Contributable Business or Life Activities Using Conformance Checking“. In: *Procedia Computer Science* 207 (2022). Knowledge-Based and Intelligent Information & Engineering Systems: Proceedings of the 26th International Conference KES-2022, S. 542–551. ISSN: 1877-0509. DOI: https://doi.org/10.1016/j.procs.2022.09.109. URL: https://www.sciencedirect.com/science/article/pii/S1877050922009905.

[83] Lucianna Kiffer, Asad Salman, Dave Levin, Alan Mislove und Cristina Nita-Rotaru. „Under the Hood of the Ethereum Gossip Protocol“. In: *Financial Cryptography and Data Security*. Hrsg. von Nikita Borisov und Claudia Diaz. Berlin, Heidelberg: Springer Berlin Heidelberg, 2021, S. 437–456. ISBN: 978-3-662-64331-0.

[84] Philipp Klinger und Freimut Bodendorf. „Blockchain-based Cross-Organizational Execution Framework for Dynamic Integration of Process Collaborations“. In: *Entwicklungen, Chancen und Herausforderungen der Digitalisierung: Proceedings der 15. Internationalen Tagung Wirtschaftsinformatik, WI 2020, Potsdam, Germany, March 9-11, 2020. Zentrale Tracks.* Hrsg. von Norbert Gronau, Moreen Heine, Hanna Krasnova und K. Poustcchi. 2020. URL: https://api.semanticscholar.org/CorpusID:215923915.

[85] Philipp Klinger, Long Nguyen und Freimut Bodendorf. „Upgradeability Concept for Collaborative Blockchain-Based Business Process Execution Framework“. In: *Blockchain – ICBC 2020*. Hrsg. von Zhixiong Chen, Laizhong Cui, Balaji Palanisamy und Liang-Jie Zhang. Cham: Springer International Publishing, 2020, S. 127–141. ISBN: 978-3-030-59638-5.

[86] Christopher Klinkmüller, Alexander Ponomarev, An Binh Tran, Ingo Weber und Wil van der Aalst. „Mining Blockchain Processes: Extracting Process Mining Data from Blockchain Applications“. In: *Business Process Management: Blockchain and Central*

and Eastern Europe Forum. Hrsg. von Claudio Di Ciccio, Renata Gabryelczyk, Luciano García-Bañuelos, Tomislav Hernaus, Rick Hull, Mojca Indihar Štemberger, Andrea Kő und Mark Staples. Cham: Springer International Publishing, 2019, S. 71–86. ISBN: 978-3-030-30429-4.

[87] Julius Köpke. „Towards Modeling Privity and Enforceability Requirements for BPM based Smart Contracts (invited paper)“. In: *Modellierung*. 2020. URL: https://api.semanticscholar.org/CorpusID:211830616.

[88] Julius Köpke, Marco Franceschetti und Johann Eder. „Balancing Privity and Enforceability of BPM-Based Smart Contracts on Blockchains“. In: *Business Process Management: Blockchain and Central and Eastern Europe Forum*. Hrsg. von Claudio Di Ciccio, Renata Gabryelczyk, Luciano García-Bañuelos, Tomislav Hernaus, Rick Hull, Mojca Indihar Štemberger, Andrea Kő und Mark Staples. Cham: Springer International Publishing, 2019, S. 87–102. ISBN: 978-3-030-30429-4.

[89] Michał Kowalski, Zach W.Y. Lee und Tommy K.H. Chan. „Blockchain technology and trust relationships in trade finance“. In: *Technological Forecasting and Social Change* 166 (2021), S. 120641. ISSN: 0040-1625. DOI: https://doi.org/10.1016/j.techfore.2021.120641. URL: https://www.sciencedirect.com/science/article/pii/S0040162521000731.

[90] Winfried Krieger. *Gabler Wirtschaftslexikon: Collaboration*. 2018. URL: https://wirtschaftslexikon.gabler.de/definition/collaboration-51807/version-274958 (besucht am 24. 11. 2023).

[91] Akhil Kumar, Rong Liu und Zhe Shan. „Is Blockchain a Silver Bullet for Supply Chain Management? Technical Challenges and Research Opportunities“. In: *Decision Sciences* 51.1 (2020), S. 8–37. DOI: https://doi.org/10.1111/deci.12396. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/deci.12396. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/deci.12396.

[92] Julius Köpke, Giovanni Meroni und Mattia Salnitri. „Designing secure business processes for blockchains with SecBPMN2BC“. In: *Future Generation Computer Systems* 141 (2023), S. 382–398. ISSN: 0167-739X. DOI: https://doi.org/10.1016/j.future.2022.11.013. URL: https://www.sciencedirect.com/science/article/pii/S0167739X22003764.

[93] Jan Ladleif. „Enforceability aspects of smart contracts on blockchain networks“. doctoralthesis. Universität Potsdam, 2021, S. xix, 152. DOI: 10.25932/publishup-51908.

[94] Jan Ladleif, Christian Friedow und Mathias Weske. „An Architecture for Multi-chain Business Process Choreographies“. In: *Business Information Systems*. Hrsg. von Witold Abramowicz und Gary Klein. Cham: Springer International Publishing, 2020, S. 184–196. ISBN: 978-3-030-53337-3.

[95] Jan Ladleif und Mathias Weske. „Time in Blockchain-Based Process Execution“. In: *2020 IEEE 24th International Enterprise Distributed Object Computing Conference (EDOC)*. 2020, S. 217–226. DOI: 10.1109/EDOC49727.2020.00034.

[96] Jan Ladleif, Mathias Weske und Ingo Weber. „Modeling and Enforcing Blockchain-Based Choreographies“. In: *Business Process Management: 17th International Conference, BPM 2019, Vienna, Austria, September 1–6, 2019, Proceedings*. Vienna, Austria: Springer-Verlag, 2019, 69–85. ISBN: 978-3-030-26618-9. DOI: 10.1007/978-3-030-26619-6_7. URL: https://doi.org/10.1007/978-3-030-26619-6_7.

[97] Leslie Lamport. „Time, Clocks, and the Ordering of Events in a Distributed System“. In: *Commun. ACM* 21.7 (1978), 558–565. ISSN: 0001-0782. DOI: 10.1145/359545.359563. URL: https://doi.org/10.1145/359545.359563.

[98] Leslie Lamport. „Using Time Instead of Timeout for Fault-Tolerant Distributed Systems.“ In: *ACM Trans. Program. Lang. Syst.* 6.2 (1984), 254–280. ISSN: 0164-0925. DOI: 10.1145/2993.2994. URL: https://doi.org/10.1145/2993.2994.

[99] Leslie Lamport, Robert Shostak und Marshall Pease. „The Byzantine Generals Problem“. In: *ACM Trans. Program. Lang. Syst.* 4.3 (1982), 382–401. ISSN: 0164-0925. DOI: 10.1145/357172.357176. URL: https://doi.org/10.1145/357172.357176.

[100] Bahareh Lashkari und Petr Musilek. „A Comprehensive Review of Blockchain Consensus Mechanisms“. In: *IEEE Access* 9 (2021), S. 43620–43652. DOI: 10.1109/ACCESS.2021.3065880.

[101] Christine Legner und Kristin Wende. „The Challenges of Inter-Organizational Business Process Design - A Research Agenda“. In: *European Conference on Information Systems*. 2007. URL: https://api.semanticscholar.org/CorpusID:14282283.

[102] Xiaoqi Li, Peng Jiang, Ting Chen, Xiapu Luo und Qiaoyan Wen. „A survey on the security of blockchain systems“. In: *Future Generation Computer Systems* 107 (2020), S. 841–853. ISSN: 0167-739X. DOI: https://doi.org/10.1016/j.future.2017.08.020. URL: https://www.sciencedirect.com/science/article/pii/S0167739X17318332.

[103] Tom Lichtenstein, Hassan Atwi, Mathias Weske und Cesare Pautasso. „Loose Collaborations on the Blockchain: Survey and Challenges“. In: *Business Process Management: Blockchain, Robotic Process Automation and Educators Forum*. Hrsg. von Julius Köpke, Orlenys López-Pintado, Ralf Plattfaut, Jana-Rebecca Rehse, Katarzyna Gdowska, Fernanda Gonzalez-Lopez, Jorge Munoz-Gama, Koen Smit und Jan Martijn E. M. van der Werf. Cham: Springer Nature Switzerland, 2023, S. 21–35. ISBN: 978-3-031-43433-4.

[104] Tom Lichtenstein, Simon Siegert, Adriatik Nikaj und Mathias Weske. „Data-Driven Process Choreography Execution on the Blockchain: A Focus on Blockchain Data Reusability“. In: *Business Information Systems*. Hrsg. von Witold Abramowicz und Gary Klein. Cham: Springer International Publishing, 2020, S. 224–235. ISBN: 978-3-030-53337-3.

[105] Frank Lindert und Wolfgang Deiters. „Modelling Inter-Organizational Processes with Process Model Fragments“. In: *Enterprise-wide and Cross-enterprise Workflow Management*. 1999. URL: https://api.semanticscholar.org/CorpusID:1660308.

[106] Chengfei Liu, Qing Li und Xiaohui Zhao. „Challenges and opportunities in collaborative business process management: Overview of recent advances and introduction to the special issue“. In: *Information Systems Frontiers* 11.3 (2009), S. 201–209. ISSN: 1572-9419. DOI: 10.1007/s10796-008-9089-0. URL: https://doi.org/10.1007/s10796-008-9089-0.

[107] Orlenys López-Pintado, Marlon Dumas, Luciano García-Bañuelos und Ingo Weber. „Controlled Flexibility in Blockchain-Based Collaborative Business Processes“. In: *Inf. Syst.* 104.C (2022). ISSN: 0306-4379. DOI: 10.1016/j.is.2020.101622. URL: https://doi.org/10.1016/j.is.2020.101622.

[108] Orlenys López-Pintado, Marlon Dumas, Luciano García-Bañuelos und Ingo Weber. „Dynamic Role Binding in Blockchain-Based Collaborative Business Processes“. In: *Advanced Information Systems Engineering*. Hrsg. von Paolo Giorgini und Barbara Weber. Cham: Springer International Publishing, 2019, S. 399–414. ISBN: 978-3-030-21290-2.

[109] Orlenys López-Pintado, Marlon Dumas, Luciano García-Bañuelos und Ingo Weber. „Interpreted Execution of Business Process Models on Blockchain“. In: *2019 IEEE 23rd International Enterprise Distributed Object Computing Conference (EDOC)* (2019), S. 206–215. URL: https://api.semanticscholar.org/CorpusID:174798045.

[110] Orlenys López-Pintado, Luciano García-Bañuelos, Marlon Dumas und Ingo Weber. „Caterpillar: A Blockchain-Based Business Process Management System“. In: *International Conference on Business Process Management*. 2017. URL: https://api.se\-manticscholar.org/CorpusID:3629238.

[111] Faiza Loukil, Khouloud Boukadi, Mourad Abed und Chirine Ghedira-Guegan. „Decentralized collaborative business process execution using blockchain“. In: *World Wide Web* 24.5 (2021), S. 1645–1663. ISSN: 1573-1413. DOI: 10.1007/s11280-021-00901-7. URL: https://doi.org/10.1007/s11280-021-00901-7.

[112] Qinghua Lu, An Binh Tran, Ingo Weber, Hugo O'Connor, Paul Rimba, Xiwei Xu, Mark Staples, Liming Zhu und Ross Jeffery. „Integrated model-driven engineering of blockchain applications for business processes and asset management“. In: *Software: Practice and Experience* 51.5 (2021), S. 1059–1079. DOI: https://doi.org/10.1002/spe.2931. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/spe.2931. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/spe.2931.

[113] Orlenys López-Pintado, Luciano García-Bañuelos, Marlon Dumas, Ingo Weber und Alexander Ponomarev. „Caterpillar: A business process execution engine on the Ethereum blockchain“. In: *Software: Practice and Experience* 49.7 (2019), S. 1162–1193. DOI: https://doi.org/10.1002/spe.2702. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/spe.2702. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/spe.2702.

[114] Morales-Sandoval M, Molina JA, Marin-Castro HM und Gonzalez-Compean JL. „Blockchain support for execution, monitoring and discovery of inter-organizational business processes“. In: *PeerJ Computer Science* (2021). DOI: https://doi.org/10.7717/peerj-cs.731.

[115] Mads Frederik Madsen, Mikkel Gaub, Tróndur Høgnason, Malthe Ettrup Kirkbro, Ettrup, Slaats Tijs, Debois, Søren und Søren Debois. „Collaboration among Adversaries Distributed Workflow Execution on a Blockchain“. In: 2018. URL: https://api.semanticscholar.org/CorpusID:215764509.

[116] Salvatore T. March und Gerald F. Smith. „Design and natural science research on information technology“. In: *Decision Support Systems* 15.4 (1995), S. 251–266. ISSN: 0167-9236. DOI: https://doi.org/10.1016/0167-9236(94)00041-2. URL: https://www.sciencedirect.com/science/article/pii/0167923694000412.

[117] Daniel Martin, Daniel Wutke und Frank Leymann. „A Novel Approach to Decentralized Workflow Enactment“. In: *2008 12th International IEEE Enterprise Distributed Object Computing Conference*. 2008, S. 127–136. DOI: 10.1109/EDOC.2008.22.

[118] Jan Mendling. „Event-Driven Process Chains (EPC)“. In: *Metrics for Process Models: Empirical Foundations of Verification, Error Prediction, and Guidelines for Correctness*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, S. 17–57. ISBN: 978-3-540-89224-3. DOI: 10.1007/978-3-540-89224-3_2. URL: https://doi.org/10.1007/978-3-540-89224-3_2.

[119] Jan Mendling. „Towards Blockchain Support for Business Processes“. In: *Business Modeling and Software Design*. Hrsg. von Boris Shishkov. Cham: Springer International Publishing, 2018, S. 243–248. ISBN: 978-3-319-94214-8.

[120] Jan Mendling u. a. „Blockchains for Business Process Management - Challenges and Opportunities“. In: *ACM Trans. Manage. Inf. Syst.* 9.1 (2018). ISSN: 2158-656X. DOI: 10.1145/3183367. URL: https://doi.org/10.1145/3183367.

[121] Alfred J. Menezes, Paul C. van Oorschot und Scott A. Vanstone. *Handbook of Applied Cryptography*. CRC Press, 2001.

[122] Lucie Mercenne, Kei-Leo Brousmiche und Elyes Ben Hamida. „Blockchain Studio: A Role-Based Business Workflows Management System“. In: *2018 IEEE 9th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*. 2018, S. 1215–1220. DOI: 10.1109/IEMCON.2018.8614879.

[123] Ralph Charles Merkle. „Secrecy, Authentication, and Public Key Systems.“ AAI8001972. Diss. Stanford, CA, USA, 1979.

[124] Giovanni Meroni, Pierluigi Plebani und Francesco Vona. „Trusted Artifact-Driven Process Monitoring of Multi-party Business Processes with Blockchain“. In: *Business Process Management: Blockchain and Central and Eastern Europe Forum*. Hrsg. von Claudio Di Ciccio, Renata Gabryelczyk, Luciano García-Bañuelos, Tomislav Hernaus, Rick Hull, Mojca Indihar Štemberger, Andrea Kő und Mark Staples. Cham: Springer International Publishing, 2019, S. 55–70. ISBN: 978-3-030-30429-4.

[125] Michael Merz, Boris Liberman, Kai Müller-Jones und Winfried Lamersdorf. „Interorganizational Workflow Managment with Mobile Agents in COSM“. In: *Proceedings of the First International Conference on the Practical Application of Intelligent Agents and Multi-Agent Technology, PAAM 1996, Westminster Central Hall, London, UK, April 22-24, 1996*. Hrsg. von I. Barry Crabtree und Nick R. Jennings. Practical Application Company Ltd., 1996, S. 405–420.

[126] Gianmaria Del Monte, Diego Pennino und Maurizio Pizzonia. „Scaling Blockchains without Giving up Decentralization and Security: A Solution to the Blockchain Scalability Trilemma“. In: *Proceedings of the 3rd Workshop on Cryptocurrencies and Blockchains for Distributed Systems*. CryBlock '20. London, United Kingdom: Association for Computing Machinery, 2020, 71–76. ISBN: 9781450380799. DOI: 10.1145/3410699.3413800. URL: https://doi.org/10.1145/3410699.3413800.

[127] Roman Mühlberger, Stefan Bachhofner, Eduardo Castelló Ferrer, Claudio Di Ciccio, Ingo Weber, Maximilian Wöhrer und Uwe Zdun. „Foundational Oracle Patterns: Connecting Blockchain to the Off-Chain World“. In: *Business Process Management: Blockchain and Robotic Process Automation Forum*. Hrsg. von Aleksandre Asatiani, José María García, Nina Helander, Andrés Jiménez-Ramírez, Agnes Koschmider, Jan Mendling, Giovanni Meroni und Hajo A. Reijers. Cham: Springer International Publishing, 2020, S. 35–51. ISBN: 978-3-030-58779-6.

[128] Roman Mühlberger, Stefan Bachhofner, Claudio Di Ciccio, Luciano García-Bañuelos und Orlenys López-Pintado. „Extracting Event Logs for Process Mining from Data Stored on the Blockchain“. In: *Business Process Management Workshops*. Hrsg. von Chiara Di Francescomarino, Remco Dijkman und Uwe Zdun. Cham: Springer International Publishing, 2019, S. 690–703. ISBN: 978-3-030-37453-2.

[129] Marcel Müller, Nadine Ostern und Michael Rosemann. „Silver Bullet for All Trust Issues? Blockchain-Based Trust Patterns for Collaborative Business Processes“. In: *Business Process Management: Blockchain and Robotic Process Automation Forum*. Hrsg. von Aleksandre Asatiani, José María García, Nina Helander, Andrés Jiménez-Ramírez, Agnes Koschmider, Jan Mendling, Giovanni Meroni und Hajo A. Reijers. Cham: Springer International Publishing, 2020, S. 3–18. ISBN: 978-3-030-58779-6.

[130] Siguna Müller. „On the Computation of Square Roots in Finite Fields“. In: *Designs, Codes and Cryptography* 31.3 (2004), S. 301–312. ISSN: 1573-7586. DOI: 10.1023/B:DESI.0000015890.44831.e2. URL: https://doi.org/10.1023/B:DESI.0000015890.44831.e2.

[131] Hirofumi Nagano, Taku Shimosawa, Shimamura Atsushi und Norihisa Komoda. „Blockchain Based Cross Organizational Workflow Management System“. In: *Proceedings of the IADIS International Conference Applied Computing 2020*. Hrsg. von Hans Weghorn. 2020, S. 97–104. ISBN: 978-989-8704-24-5.

[132] Hirofumi Nagano, Taku Shimosawa, Shimamura Atsushi und Norihisa Komoda. „Reliable Architecture of Cross Organizational Workflow Management System on Blockchain“. In: *IA-*

DIS International Journal on Computer Science and Information System 15.2 (2020), S. 29–43. ISSN: 1646-3692. URL: https://www.iadisportal.org/ijcsis/papers/2020150203.pdf.

[133] Satoshi Nakamoto. *Bitcoin: A Peer-to-Peer Electronic Cash System*. 2008. URL: https://bitcoin.org/bitcoin.pdf.

[134] Hiroaki Nakamura, Kohtaroh Miyamoto und Michiharu Kudo. „Inter-organizational Business Processes Managed by Blockchain“. In: *Web Information Systems Engineering – WISE 2018*. Hrsg. von Hakim Hacid, Wojciech Cellary, Hua Wang, Hye-Young Paik und Rui Zhou. Cham: Springer International Publishing, 2018, S. 3–17. ISBN: 978-3-030-02922-7.

[135] OMG. *Business Process Model and Notation (BPMN), Version 2.0.2*. Object Management Group, Dez. 2013. URL: http://www.omg.org/spec/BPMN/2.0.2.

[136] OMG. *Decision Model and Notation*. Object Management Group, Apr. 2023. URL: https://www.omg.org/spec/DMN/1.4.

[137] Thomas Osterland, Thomas Rose und Clemens Putschli. „On the Implementation of Business Process Logic in DLT Nodes“. In: *Proceedings of the 2020 Asia Service Sciences and Software Engineering Conference*. ASSE '20. Nagoya, Japan: Association for Computing Machinery, 2020, 91–99. ISBN: 9781450377102. DOI: 10.1145/3399871.3399899. URL: https://doi.org/10.1145/3399871.3399899.

[138] Ken Peffers, Tuure Tuunanen, Marcus A. Rothenberger und Samir Chatterjee. „A Design Science Research Methodology for Information Systems Research“. In: *Journal of Management Information Systems* 24.3 (2007), S. 45–77. DOI: 10.2753/MIS0742-1222240302. URL: https://doi.org/10.2753/MIS0742-1222240302.

[139] Maja Pesic, Helen Schonenberg und Wil M.P. van der Aalst. „DECLARE: Full Support for Loosely-Structured Processes“. In: *11th IEEE International Enterprise Distributed Object Computing Conference (EDOC 2007)*. 2007, S. 287–287. DOI: 10.1109/EDOC.2007.14.

[140] Carl Adam Petri. „Kommunikation mit Automaten“. Diss. 1962.

[141] Shaya Pourmirza, Sander Peters, Remco Dijkman und Paul Grefen. „BPMS-RA: A Novel Reference Architecture for Business Process Management Systems“. In: *ACM Trans. Internet Technol.* 19.1 (2019). ISSN: 1533-5399. DOI: 10.1145/3232677. URL: https://doi.org/10.1145/3232677.

[142] A. Preukschat und D. Reed. *Self-Sovereign Identity: Decentralized Digital Identity and Verifiable Credentials*. Manning, 2021. ISBN: 9781617296598. URL: https://books.google.de/books?id=Nh4uEAAAQBAJ.

[143] Christoph Prybila, Stefan Schulte, Christoph Hochreiner und Ingo Weber. „Runtime verification for business processes utilizing the Bitcoin blockchain“. In: *Future Generation Computer Systems* 107 (2020), S. 816–831. ISSN: 0167-739X. DOI: https://doi.org/10.1016/j.future.2017.08.024. URL: https://www.sciencedirect.com/science/article/pii/S0167739X1731837X.

[144] Manfred Reichert und Barbara Weber. *Enabling Flexibility in Process-Aware Information Systems*. Springer Berlin, Heidelberg, 2012. ISBN: 978-3-642-30409-5. DOI: 10.1007/978-3-642-30409-5.

[145] Olivier Rikken, Marijn Janssen und Zenlin Kwee. „The ins and outs of decentralized autonomous organizations (DAOs) unraveling the definitions, characteristics, and emerging developments of DAOs“. In: *Blockchain: Research and Applications* 4.3 (2023), S. 100143. ISSN: 2096-7209. DOI: https://doi.org/10.1016/j.bcra.2023.100143. URL: https://www.sciencedirect.com/science/article/pii/S2096720923000180.

[146] R. L. Rivest, A. Shamir und L. Adleman. „A Method for Obtaining Digital Signatures and Public-Key Cryptosystems“. In: *Commun. ACM* 21.2 (1978). ISSN: 0001-0782. DOI: 10.1145/359340.359342.

[147] Jeremy Roschelle und Stephanie D. Teasley. „The Construction of Shared Knowledge in Collaborative Problem Solving“. In: *Computer Supported Collaborative Learning*. Hrsg. von Claire O'Malley. Berlin, Heidelberg: Springer Berlin Heidelberg, 1995, S. 69–97. ISBN: 978-3-642-85098-1.

[148] Nick Russell, Wil M. P. van der Aalst, Arthur H. M. ter Hofstede und David Edmond. „Workflow Resource Patterns: Identification, Representation and Tool Support“. In: *Advanced Information Systems Engineering*. Hrsg. von Oscar Pastor und João Falcão e Cunha. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, S. 216–232. ISBN: 978-3-540-32127-9.

[149] Nirmiti Sali, Srushti Biwalkar, Amisha Swamy, Nimisha Khadilkar und Sunita Sahu. „Blockchain in Food Supply Chain: Benefits, Challenges and Review“. In: *International Journal for Research in Applied Science and Engineering Technology* 11 (Apr. 2023), S. 4654–4661. DOI: 10.22214/ijraset.2023.50661.

[150] Cyril Naves Samuel, Severine Glock, François Verdier und Patricia Guitton-Ouhamou. „Choice of Ethereum Clients for Private Blockchain: Assessment from Proof of Authority Perspective“. In: *2021 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*. 2021, S. 1–5. DOI: 10.1109/ICBC51069.2021.9461085.

[151] Abdurrashid Ibrahim Sanka und Ray C.C. Cheung. „A systematic review of blockchain scalability: Issues, solutions, analysis and future research“. In: *Journal of Network and Computer Applications* 195 (2021), S. 103232. ISSN: 1084-8045. DOI: https://doi.org/10.1016/j.jnca.2021.103232. URL: https://www.sciencedirect.com/science/article/pii/S1084804521002307.

[152] Markus Schinle, Christina Erler, Philip Nicolai Andris und Wilhelm Stork. „Integration, Execution and Monitoring of Business Processes with Chaincode“. In: *2020 2nd Conference on Blockchain Research & Applications for Innovative Networks and Services (BRAINS)*. 2020, S. 63–70. DOI: 10.1109/BRAINS49436.2020.9223283.

[153] Stefan Schönig, Lars Ackermann, Stefan Jablonski und Andreas Ermer. „IoT meets BPM: a bidirectional communication architecture for IoT-aware process execution“. In: *Software and Systems Modeling* 19.6 (2020), S. 1443–1459. ISSN: 1619-1374. DOI: 10.1007/s10270-020-00785-7. URL: https://doi.org/10.1007/s10270-020-00785-7.

[154] Georg Schreyögg. *Grundlagen der Organisation: Basiswissen für Studium und Praxis*. Wiesbaden: Springer Gabler, 2020. ISBN: 978-3-658-13959-9. DOI: https://doi.org/10.1007/978-3-658-13959-9.

[155] Michael L. Scott. *Programming Language Pragmatics*. 4. Aufl. Morgan Kaufmann, 2015. ISBN: 978-0-12-410409-9.

[156] Johannes Sedlmeir, Jonathan Lautenschlager, Gilbert Fridgen und Nils Urbach. „The transparency challenge of blockchain in organizations“. In: *Electronic Markets* 32.3 (2022), S. 1779–1794. ISSN: 1422-8890. DOI: 10.1007/s12525-022-00536-0. URL: https://doi.org/10.1007/s12525-022-00536-0.

[157] Maung K. Sein, Ola Henfridsson, Sandeep Purao, Matti Rossi und Rikard Lindgren. „Action Design Research“. In: *MIS Quarterly* 35.1 (2011), S. 37–56. ISSN: 02767783. URL: http://www.jstor.org/stable/23043488.

[158] Michelle Shumate, Yannick Atouba, Katherine R. Cooper und Andrew Pilny. „Interorganizational Communication“. In: *The International Encyclopedia of Organizational Communication*. John Wiley & Sons, Ltd, 2016, S. 1–24. ISBN: 9781118955567. DOI: https://doi.org/10.1002/9781118955567.wbieoc117. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/9781118955567.wbieoc117. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/9781118955567.wbieoc117.

[159] Diogo Silva, Sérgio Guerreiro und Pedro Sousa. „Decentralized Enforcement of Business Process Control Using Blockchain“. In: *Advances in Enterprise Engineering XII*. Hrsg. von

David Aveiro, Giancarlo Guizzardi, Sérgio Guerreiro und Wided Guédria. Cham: Springer International Publishing, 2019, S. 69–87. ISBN: 978-3-030-06097-8.

[160] Victor Amaral de Sousa, Corentin Burnay und Monique Snoeck. „Process-Aware and Shared Domain Modeling for Blockchain-Enabled Business Processes: The B-Merode Language“. In: *SSRN Electronic Journal* (2022). URL: https://api.semanticscholar.org/CorpusID:249168020.

[161] Herbert Stachowiak. *Allgemeine Modelltheorie*. Springer Wien, 1973.

[162] Maarten van Steen und Andrew Tanenbaum. *Distributed Systems*. 4th. Maarten van Steen, 2024. ISBN: 978-90-815406-4-3.

[163] Fabian Stiehle. „Next-Generation Trusted Process Enactment using Blockchain State Channels“. In: *Proceedings of the Best Dissertation Award, Doctoral Consortium, and Demonstration & Resources Forum at BPM 2023 co-located with 21st International Conference on Business Process Management (BPM 2023), Utrecht, The Netherlands, September 11th to 15th, 2023*. Hrsg. von Dirk Fahland u. a. Bd. 3469. CEUR Workshop Proceedings. CEUR-WS.org, 2023, S. 16–22. URL: https://ceur-ws.org/Vol-3469/paper-04.pdf.

[164] Fabian Stiehle und Ingo Weber. „Blockchain for Business Process Enactment: A Taxonomy and Systematic Literature Review“. In: *Business Process Management: Blockchain, Robotic Process Automation, and Central and Eastern Europe Forum*. Hrsg. von Andrea Marrella, Raimundas Matulevičius, Renata Gabryelczyk, Bernhard Axmann, Vesna Bosilj Vukšić, Walid Gaaloul, Mojca Indihar Štemberger, Andrea Kő und Qinghua Lu. Cham: Springer International Publishing, 2022, S. 5–20. ISBN: 978-3-031-16168-1.

[165] Jan Vanthienen Stijn Goedertier und Filip Caron. „Declarative business process modelling: principles and modelling languages“. In: *Enterprise Information Systems* 9.2 (2015), S. 161–185. DOI: 10.1080/17517575.2013.830340. eprint: https://doi.org/10.1080/17517575.2013.830340. URL: https://doi.org/10.1080/17517575.2013.830340.

[166] Christian Sturm und Stefan Jablonski. „Interorganizational Process Execution Beyond Ethereum: Road to a Special Purpose Ecosystem“. In: *PoEM Workshops*. 2020. URL: https://api.semanticscholar.org/CorpusID:229623402.

[167] Christian Sturm und Stefan Jablonski. „The Dpex-Framework: Towards Full WFMS Support for Decentralized Process Execution“. In: *Business Process Management Forum*. Hrsg. von Chiara Di Francescomarino, Andrea Burattin, Christian Janiesch und

Shazia Sadiq. Cham: Springer Nature Switzerland, 2023, S. 20–37. ISBN: 978-3-031-41623-1.

[168] Christian Sturm, Jonas Scalanczi, Stefan Schönig und Stefan Jablonski. „A Blockchain-based and resource-aware process execution engine“. In: *Future Generation Computer Systems* 100 (2019), S. 19–34. ISSN: 0167-739X. DOI: https://doi.org/10.1016/j.future.2019.05.006. URL: https://www.sciencedirect.com/science/article/pii/S0167739X18327158.

[169] Christian Sturm, Jonas Szalanczi, Stefan Jablonski und Stefan Schönig. „Decentralized Control: A Novel Form of Interorganizational Workflow Interoperability“. In: *The Practice of Enterprise Modeling*. Hrsg. von Jānis Grabis und Dominik Bork. Cham: Springer International Publishing, 2020, S. 261–276. ISBN: 978-3-030-63479-7.

[170] Christian Sturm, Jonas Szalanczi, Stefan Schönig und Stefan Jablonski. „A Lean Architecture for Blockchain Based Decentralized Process Execution“. In: *Business Process Management Workshops*. Hrsg. von Florian Daniel, Quan Z. Sheng und Hamid Motahari. Cham: Springer International Publishing, 2019, S. 361–373. ISBN: 978-3-030-11641-5.

[171] Ahmed T. Suliman, Maha Kadadha, Rabeb Mizouni, Hadi Otrok, Ernesto Damiani und Mahmoud Al-Qutayri. „Blockcheck: A consortium blockchain-based conformance checking framework for business processes“. In: *Internet of Things* 21 (2023), S. 100652. ISSN: 2542-6605. DOI: https://doi.org/10.1016/j.iot.2022.100652. URL: https://www.sciencedirect.com/science/article/pii/S2542660522001330.

[172] Satyananda Swain, und Manas Ranjan Patra. „An Inclusive Smart Contract Deployment for the Procurement Cycle in a Software Agent-Oriented Blockchain-Enabled Supply Chain Framework“. In: *Research Journal of Berhampur University* V (Juni 2023). Hrsg. von Pratap Kumar Mohanty. ISSN: 2250-168.

[173] Nick Szabo. „Formalizing and Securing Relationships on Public Networks“. In: *First Monday* 2.9 (1997). DOI: 10.5210/fm.v2i9.548. URL: https://firstmonday.org/ojs/index.php/fm/article/view/548.

[174] Paolo Tasca und Claudio J. Tessone. „A Taxonomy of Blockchain Technologies: Principles of Identification and Classification“. In: *Ledger* 4 (2019). DOI: 10.5195/ledger.2019.140. URL: https://ledger.pitt.edu/ojs/ledger/article/view/140.

[175] Rodrigue Tonga Naha und Kaiwen Zhang. „Pupa: Smart Contracts for BPMN with Time-Dependent Events and Inclusive Gateways“. In: *Business Process Management: Blockchain, Robotic Process Automation, and Central and Eastern Europe Forum*.

Hrsg. von Andrea Marrella, Raimundas Matulevičius, Renata Gabryelczyk, Bernhard Axmann, Vesna Bosilj Vukšić, Walid Gaaloul, Mojca Indihar Štemberger, Andrea Kő und Qinghua Lu. Cham: Springer International Publishing, 2022, S. 21–35. ISBN: 978-3-031-16168-1.

[176] An Binh Tran, Qinghua Lu und Ingo Weber. „Lorikeet: A Model-Driven Engineering Tool for Blockchain-Based Business Process Execution and Asset Management“. In: *International Conference on Business Process Management*. 2018. URL: https://api.semanticscholar.org/CorpusID:52195200.

[177] Wattana Viriyasitavat, Li Da Xu, Gaurav Dhiman und Zhuming Bi. „Blockchain-as-a-Service for Business Process Management: Survey and Challenges“. In: *IEEE Transactions on Services Computing* 16.3 (2023), S. 2299–2314. DOI: 10.1109/TSC.2022.3199232.

[178] Wattana Viriyasitavat, Li Da Xu, Dusit Niyato, Zhuming Bi und Danupol Hoonsopon. „Applications of Blockchain in Business Processes: A Comprehensive Review“. In: *IEEE Access* 10 (2022), S. 118900–118925. DOI: 10.1109/ACCESS.2022.3217794.

[179] Markus Voelter. *DSL Engineering*. 2013. URL: http://voelter.de/dslbook/markusvoelter-dslengineering-1.0.pdf.

[180] Marko Vukolić. „The Quest for Scalable Blockchain Fabric: Proof-of-Work vs. BFT Replication“. In: *Open Problems in Network Security*. Hrsg. von Jan Camenisch und Doğan Kesdoğan. Cham: Springer International Publishing, 2016, S. 112–125. ISBN: 978-3-319-39028-4.

[181] Stefan Wagner. *Software Product Quality Control*. 1. Aufl. Springer Berlin, Heidelberg, 2013. ISBN: 978-3-642-38571-1. DOI: 10.1007/978-3-642-38571-1.

[182] Hong Wan, Kejun Li und Yining Huang. „Blockchain: A Review from The Perspective of Operations Researchers“. In: *2020 Winter Simulation Conference (WSC)*. 2020, S. 75–89. DOI: 10.1109/WSC48552.2020.9383924.

[183] Puwei Wang, Zhouxing Sun, Rui Li, Jinchuan Chen, Ping Gong und Xiaoyong Du. „An Efficient Customized Blockchain System for Inter-Organizational Processes“. In: *2023 IEEE International Conference on Web Services (ICWS)*. 2023, S. 615–625. DOI: 10.1109/ICWS60048.2023.00080.

[184] Yina Wang, Hongbin Ma, Qitao Ma, Hong Chen, Dongdong Zhang und Yingli Wang. „Block-Based Data Security Storage Scheme“. In: *Communications, Signal Processing, and Systems*. Hrsg. von Qilian Liang, Wei Wang, Xin Liu, Zhenyu Na, Min Jia und Baoju Zhang. Singapore: Springer Singapore, 2020, S. 1567–1575. ISBN: 978-981-13-9409-6.

[185] Ingo Weber, Xiwei Xu, Régis Riveret, Guido Governatori, Alexander Ponomarev und Jan Mendling. „Untrusted Business Process Monitoring and Execution Using Blockchain“. In: *Business Process Management*. Hrsg. von Marcello La Rosa, Peter Loos und Oscar Pastor. Cham: Springer International Publishing, 2016, S. 329–347. ISBN: 978-3-319-45348-4.

[186] Jan Werth., Nabil El Ioini., Mohammad Berenjestanaki., Hamid Barzegar. und Claus Pahl. „A Platform Selection Framework for Blockchain-Based Software Systems Based on the Blockchain Trilemma“. In: *Proceedings of the 18th International Conference on Evaluation of Novel Approaches to Software Engineering - ENASE*. INSTICC. SciTePress, 2023, S. 362–371. ISBN: 978-989-758-647-7. DOI: 10.5220/0011837300003464.

[187] Mathias Weske. *Business Process Management*. 3. Aufl. Springer Berlin, Heidelberg, 2019. ISBN: 978-3-662-59432-2. DOI: 10.1007/978-3-662-59432-2.

[188] Claes Wohlin. „Guidelines for Snowballing in Systematic Literature Studies and a Replication in Software Engineering“. In: *Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering*. EASE '14. London, England, United Kingdom: Association for Computing Machinery, 2014. ISBN: 9781450324762. DOI: 10.1145/2601248.2601268. URL: https://doi.org/10.1145/2601248.2601268.

[189] Gavin Wood. *Ethereum: A Secure Decentralised Generalised Transaction Ledger*. Techn. Ber. London Version 818564b. Ethereum, Nov. 2023.

[190] Gavin Wood. *Ethereum: A Secure Decentralised Generalised Transaction Ledger*. Techn. Ber. Paris Version 2f36cf0. Ethereum, Jan. 2024.

[191] Bastian Wunderlich. „Adaption der User-Interfaces von Workflow Management Systemen zur Steuerung des dpex-Frameworks für die dezentrale Prozessausführung am Beispiel Camunda“. Bachelorarbeit (unveröffentlicht). Universität Bayreuth, 30. Jan. 2024.

[192] Yibin Xu, Tijs Slaats, Boris Düdder, Søren Debois und Haiqin Wu. „Distributed and Adversarial Resistant Workflow Execution on the Algorand Blockchain“. In: *Financial Cryptography and Data Security. FC 2022 International Workshops*. Hrsg. von Shin'ichiro Matsuo, Lewis Gudgeon, Ariah Klages-Mundt, Daniel Perez Hernandez, Sam Werner, Thomas Haines, Aleksander Essex, Andrea Bracciali und Massimiliano Sala. Cham: Springer International Publishing, 2023, S. 583–597. ISBN: 978-3-031-32415-4.

[193] Yibin Xu, Tijs Slaats und Boris Düdder. „A two-dimensional sharding model for access control and data privilege management of blockchain“. In: *Simulation Modelling Practice and Theory* 122 (2023), S. 102678. ISSN: 1569-190X. DOI: https://doi.org/10.1016/j.simpat.2022.102678. URL: https://www.sciencedirect.com/science/article/pii/S1569190X22001484.

[194] Dongjin YU, Yijie WEI, Xiaoxiao SUN, Ke NI und Hujun SHEN. „Choreography-driven business process management framework based on blockchain“. In: *Journal on Communications* 42 (9 2021), S. 120–132.

[195] Johannes Maria Zaha, Alistair Barros, Marlon Dumas und Arthur ter Hofstede. „Let's Dance: A Language for Service Behavior Modeling“. In: *On the Move to Meaningful Internet Systems 2006: CoopIS, DOA, GADA, and ODBASE*. Hrsg. von Robert Meersman und Zahir Tari. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, S. 145–162. ISBN: 978-3-540-48289-5.

[196] Milliam Maxime Zekeng Ndadji, Maurice Tchoupé Tchendji, Clémentin Tayou Djamegni und Didier Parigot. „A Language and Methodology based on Scenarios, Grammars and Views, for Administrative Business Processes Modelling“. In: *ParadigmPlus* 1.3 (2020), S. 1–22. DOI: 10.55969/paradigmplus.v1n3a1. URL: https://journals.itiud.org/index.php/paradigmplus/article/view/15.

[197] Dawen Zhang, Xiwei Xu, Liming Zhu und Hye-Young Paik. „A Process Adaptation Framework for Blockchain-Based Supply Chain Management“. In: *2021 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*. 2021, S. 1–9. DOI: 10.1109/ICBC51069.2021.9461106.

[198] Fangguo Zhang und Kwangjo Kim. „ID-Based Blind Signature and Ring Signature from Pairings“. In: *Advances in Cryptology — ASIACRYPT 2002*. Hrsg. von Yuliang Zheng. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002, S. 533–547. ISBN: 978-3-540-36178-7.

[199] Weiyu Zhong, Ce Yang, Wei Liang, Jiahong Cai, Lin Chen, Jing Liao und Naixue Xiong. „Byzantine Fault-Tolerant Consensus Algorithms: A Survey“. In: *Electronics* 12.18 (2023). ISSN: 2079-9292. DOI: 10.3390/electronics12183801. URL: https://www.mdpi.com/2079-9292/12/18/3801.

[200] W.M.P. van der Aalst und A.H.M. ter Hofstede. „YAWL: yet another workflow language“. In: *Information Systems* 30.4 (2005), S. 245–275. ISSN: 0306-4379. DOI: https://doi.org/10.1016/j.is.2004.02.002. URL: https://www.sciencedirect.com/science/article/pii/S0306437904000304.