Albert-Ludwigs-Universität Freiburg
Institut für Informatik und Gesellschaft

Dissertation zur Erlangung des
Doktorgrades der Technischen Fakultät der
Albert-Ludwigs-Universität Freiburg im Breisgau

Interdependenzen in Prozessarchitekturen

Simulation gegenseitiger Beeinflussung sicherheitsorientierter Geschäftsprozesse

vorgelegt von

Richard Markus Zahoransky

geboren in Karlsruhe

Mai 2017

Dekan:	Prof. Dr. Oliver Paul, Albert-Ludwigs-Universität Freiburg
Erstreferent:	Prof. Dr. Dr. h.c. Günter Müller, Albert-Ludwigs-Universität Freiburg
Zweitreferent:	Prof. Dr. Joachim Posegga, Universität Passau
Datum der Disputation:	03.11.2017

Danksagung

Mein Dank geht an Herrn Prof. Dr. Dr. h.c. Günter Müller, der mir diese Promotion durch seine Unterstützung ermöglicht hat. Während den Jahren stand er mir mit wertvollen Ratschlägen beiseite. Ich empfand dies nie als selbstverständlich. Seine offene Art und Diskussionsfreudigkeit ermöglichte es mir, von seinem fachlichen Wissen und Fähigkeiten zu lernen. Inspirierende Gespräche zeigten mir, dass seine Förderung über das Wissenschaftliche hinaus gingen. So verdanke ich ihm nicht nur diese Dissertation, sondern auch eine persönliche Reifung, welche für mein weiteren Lebensweg nachhaltigen Einfluss haben wird.

Herrn Prof. Dr. Joachim Posegga danke ich sehr für die Übernahme des Zweitgutachtens. Seine äußerst konstruktiven Beiträge rundeten meine Arbeit ab und ermöglichten mir das Thema genauer einzugrenzen. Seine aufgeschlossene, freundliche und unterstützende Art gab mir Mut, meine Ziele zu verfolgen.

Fachliche Unterstützung erhielt ich von Dr. Thomas Stocker, Julius Holderer und Adrian Lange. Ich danke ihnen vielmals für Ihre produktiven Kommentare und Hilfe bei meinem Vorhaben. Großen Dank geht an Dr. Thomas Koslowski für seine helfende Hand in der Wahl meines Themas. Dr. Christian Zimmermann danke ich für seine Unterstützung am Institut. Mit Dr. Christian Brenig verbrachte ich fruchtbare Zeiten mit fachlichen und privaten Diskussionen. Ich danke meinen Arbeitskollegen und Freunde, deren wertvolle Unterstützung ich erfahren durfte. Sie gaben mir fachliche Ratschläge und erlaubten mir auch Zeit zur aufbauenden Erholung. So sind es die gemeinsamen Erlebnisse, die zu Freundschaft und lebenslangen, freudigen Erinnerungen führten.

Nicht zuletzt danke ich meinen Eltern, die mich zu meinen bisherigen Lebensweg ermunterten.

Zusammenfassung

Sowohl der globale Wettbewerb als auch die sich verschärfende Gesetzeslage, setzen Unternehmen unter zunehmenden ökonomischen und regulatorischen Druck. Einerseits erfordert die Globalisierung die Effizienzsteigerung durch automatisierte Geschäftsprozesse, anderseits führten vermehrte Wirtschaftskriminalitätsdelikte zu neuen Anforderung an sichere Geschäftsprozesse aufgrund regulatorischer Vorgaben. Damit die Einhaltungen dieser Anforderungen dem unternehmerischen Effizienzstreben nicht zuwiderlaufen, wird in dieser Arbeit ein Simulationsverfahren vorgestellt, welches erstmals den Einfluss von Sicherheitskontrollen auf simultan ablaufende, gegenseitig abhängige Geschäftsprozesse aufdeckt. Dieses Simulationsverfahren stellt Zeitaspekte als zentrale Determinante in den Mittelpunkt der Betrachtung.Prozessentwickler können mit dieser Lösung die Auswirkungen ihrer Prozessänderungen a priori, vor der tatsächlichen Ausführung, prognostizieren. Dabei wird das Ziel verfolgt, eine zeit- und kostenminimierende Umsetzung der geforderten Anpassungen zu finden.

Allgemeine Trends wie Industrie 4.0 tragen zur weiteren Automatisierung von Geschäftsprozessen bei, welche so aufeinander abgestimmt sind, dass Produkte, Dienstleistungen und Waren mit hoher Taktzahl bearbeitet oder ausgeliefert werden. Prozessänderungen können wegen den wachsenden Abhängigkeiten der automatisierten Geschäftsprozesse das Zusammenspiel auf bisher unquantifizierbare Weise stören. Maßnahmen zur Stärkung der Sicherheit können deswegen von Einflussnehmern abgelehnt werden, da sich monetäre Auswirkungen auf Grund der Interdependenzen zwischen den Geschäftsprozessen bisher nicht beziffern lassen.

Das im Rahmen dieser Arbeit entwickelte Simulationsverfahren berücksichtigt die Abhängigkeiten zwischen Prozessen, deren Zugriffsrechten und die benötigte Zeit zur Abarbeitung von Einzelschritten. Dies ermöglicht die Aufdeckung der Auswirkungen einer Prozessänderung auch auf abhängige Geschäftsprozesse um die Gesamtkosten einer Änderung zu ermitteln und unterstützt hiermit Prozessentwickler, Sicherheitsmaßnahmen kosteneffizient durchzuführen. Dadurch lassen sich Bedürfnisse der Interessengruppen, die Unternehmensziele und die Sicherheit miteinander vereinen.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

ABC Activity Based Costing 27

BoD Binding of Duties 6

BPM Business Process Management 8

BPMN Business Process Model and Notation 21

BPR Business Process Re-engineering 31

CPN Colored Petri Net 48

GPM Geschäftsprozessmanagement 20

IFNet Information-Flow Petri Net 51

IKS Internes Kontrollsystem 3

ITCPN Interval Timed Colored Petri-Net 60

KPI Key Performance Indicator 27

MG Markierungsgraph 44

NAT Normal Accident Theory 11

Kapitel 1

Unternehmerisches Spannungsfeld zwischen Sicherheit und Rentabilität

Unternehmen befinden sich im Spannungsfeld zwischen ökonomischem und regulatorischem Druck (Hungenberg et al. 2015). Einerseits fordert der globale Wettbewerb die Effizienzsteigerung, zum Beispiel durch automatisierte Geschäftsprozesse, andererseits verlangt die sich verschärfende Gesetzeslage die sichere Ausgestaltung der Geschäftsprozesse. Damit die Einhaltung dieser Vorgaben die Bemühungen der Effizienzsteigerung nicht vereitelt, wird in dieser Arbeit ein Simulationsverfahren vorgestellt, welches erstmals den Einfluss von Sicherheitskontrollen auf simultan ablaufende, gegenseitig abhängige Geschäftsprozesse aufdeckt. Dieser Ansatz bietet Prozessentwicklern die Möglichkeit, bereits bei der Planung von Prozessen oder Prozessänderungen die Auswirkungen ihrer Umsetzung auf verschiedene Leistungskennzahlen zu prognostizieren und so eine bestmögliche Umsetzung von Sicherheitskontrollen mit geringer Auswirkung auf die Effizienz der Geschäftsprozesse zu finden.

Geschäftsprozesse verknüpfen Einzeltätigkeiten unter Beachtung geltender Richtlinien, um aus den unternehmenseigenen Ressourcen heraus zu einem dem Unternehmen dienlichem Ziel zu führen. Die der Globalisierung entsprungenen Trends führen zur Digitalisierung und Automatisierung der Geschäftsprozesse zur Steigerung der Wettbewerbsfähigkeit (Spath et al. 2013; Hirsch-Kreinsen 2014). Dem gegenüber stehen die in den vergangenen Jahren aufgetretenen Wirtschaftsskandale wie beispielsweise Enron oder WorldCom (Barizo 2014). Sie zeigen die Empfindlichkeit von Unternehmen aufgrund von Sicherheitsmängel ihrer Geschäftsprozesse. Im Fall

der Société Générale war es einer einzelnen Person möglich 4,9 Mrd. Euro zu veruntreuen. Weitere Skandale zeigen eine Heterogenität der Akteure. So waren es nach heutigen Vermutungen nicht die Mitarbeiter, welche den VW-Abgasskandal hervorbrachten, sondern die Konzernleitung billigte einen veränderten Ablauf der Geschäftsprozesse (Fromm et al. 2016). Dadurch wurden nicht nur regulatorische Auflagen umgangen, sondern zusätzlich gegen interne Auflagen des Unternehmens verstoßen. Dieser Skandal hinterlässt damit einen beachtlichen, monetären Verlust, als auch einen Reputations- und Imageverlust weiterer, unbeteiligter Konzerne.

Laut Studien sehen sich Unternehmen weltweit von Wirtschaftskriminalität bedroht (PwC 2016). Dieses Bild spiegelt sich auch in Deutschland wider. Der zuletzt verzeichnete Rückgang der Fallzahlen zur Wirtschaftskriminalität konnte sich nicht durchsetzen – die Fallzahlen stagnieren, siehe Abbildung 1.1. Die kumulierten Verluste durch Wirtschaftskriminalität belaufen sich für das Jahr 2015 auf 2,8 Mrd. Euro (Bundeskriminalamt 2015).

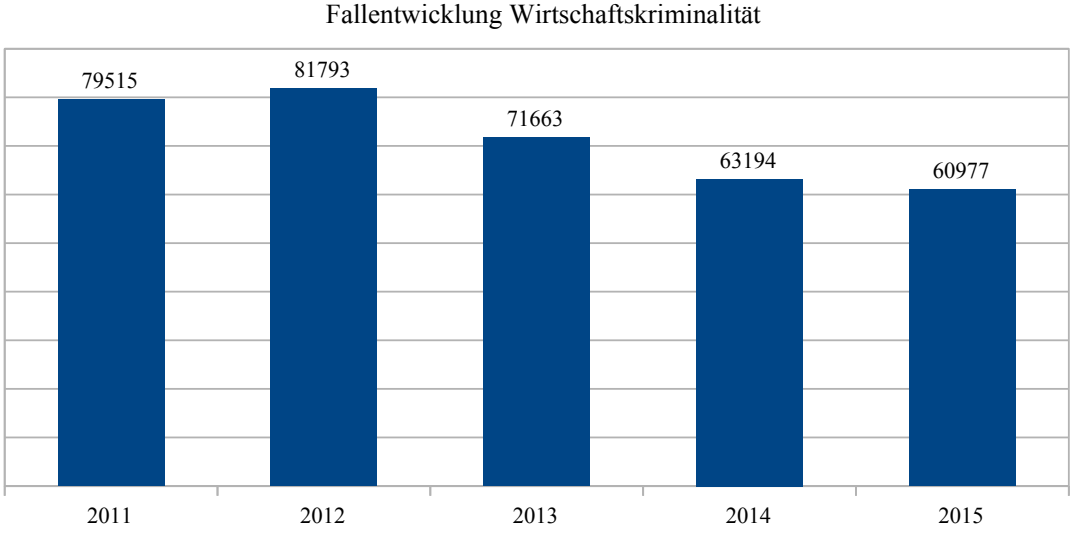


Abbildung 1.1.: Fälle von Wirtschaftskriminalität in Deutschland[1]

Der Sicherheit in Geschäftsprozessen liegt die Bekämpfung von Schwachstellen in Geschäftsprozessen, welche zu Betrugsfällen oder unerlaubten Manipulationen führen könnten, zu Grunde. Hierfür stehen technische, rechtliche und organisatorische Maßnahmen zur Verfügung.

[1] Vgl. (Bundeskriminalamt 2015)

1.1 Sicherheit in Geschäftsprozessen

Die vorgestellten Beispiele zeigen den Bedarf der sicheren Ausgestaltung, gerade von digitalen, rechnergestützten Geschäftsprozessen. Die Sicherheit in Geschäftsprozessen erweitert die Ansätze der IT-Sicherheit (Bishop 2004) durch die zusätzliche Einhaltung weiterer, abstrakterer Richtlinien und Anforderungen, welche nicht alleine durch technische Maßnahmen umzusetzen sind (Kaspersky Lab 2016). Unternehmen sind durch die allgemeine Sensibilisierung für Wirtschaftskriminalität einer Vielzahl Regularien, Handlungsempfehlungen und Standards unterlegen, um die Sicherheit in Geschäftsprozessen zu erhöhen und Betrugsfälle zu minimieren oder erkennbar zu machen. Großen Einfluss hat zum Beispiel das durch den amerikanischen Gesetzgeber eingeführte SOX (Sarbanes-Oxley Act of 2002) Bundesgesetz. Es fordert die Verlässlichkeit veröffentlichter Finanzdaten durch ein Internes Kontrollsystem (IKS) mit nachgewiesener Effektivität (Alisch et al. 2013). Auch in Deutschland fand dieses Gesetzt eine entsprechende Adaptierung. Die Sicherheit von Geschäftsprozessen umfasst somit auch die Konformität zu bestehenden Auflagen. Diese haben häufig eine gemeinsame Zielsetzung – die Eindämmung oder nachträgliche Erkennbarkeit von Betrugsfällen und Schwachstellen (Claussen 2011; Pieth 2011).

Die Umsetzung und Einhaltung solcher Regularien, zusammengefasst unter dem Begriff Compliance, ist damit ebenfalls Teil der Sicherheit in Geschäftsprozessen. Aus Sicht der Unternehmen bezieht sich ein Risiko somit sowohl auf die Nichteinhaltung von Regulationen und Anforderungen als auch auf die Ausnutzung einer Sicherheitsschwachstelle durch einen Angreifer. Unter dem Begriff des Risikomanagements werden beide Ansätze zusammengefasst. Die Umsetzung des Risikomanagements, also von Sicherheit und Compliance sollte sich durch die Leit- und Kontrollprinzipien innerhalb der Grundprinzipien der Unternehmensführung – der *Corporate Governance* – unternehmensweit manifestieren.

Das Risikomanagement hat die Aufgabe, existenzgefährdende Risiken zu Identifizieren, zu bewerten und diese zu verringern (Gleißner 2011). Diese Risiken umfassen sowohl Sicherheitsmängel, als auch die Nichteinhaltung von Auflagen (Non-Compliance). Damit zielt die Sicherheit auf die Minimierung des Risikos, dass ein unerwünschtes Verhalten eintritt. Dies kann zum Beispiel ein unerlaubter Dateizugriff sein, die Nichteinhaltung einer gesetzlichen Auflage, oder das Ausführen von

Funktionen durch eine nicht berechtigte Person. Die Behandlung eines identifizierten Risikos kann verschiedenartig ausfallen:

- Ist der zu erwartende Schaden oder die Eintrittswahrscheinlichkeit gering, kann das Risiko akzeptiert werden.
- Das Risiko lässt sich durch Versicherungen auslagern.
- Die Schwachstelle, die zu dem Risiko führt, kann behandelt werden, um die Eintrittswahrscheinlichkeit oder den zu erwartenden Verlust zu verringern.

Die Kontrolle eines Risikos wird auf unterschiedlichen Ebenen innerhalb des Unternehmens realisiert, wie in Abbildung 1.2 verdeutlicht. Zum einen ist dies die Ebene der IT-Infrastruktur, welche Geschäftsprozesse realisieren lässt und unterstützt, zum anderen die organisatorische Ebene, welche die Wertschöpfung des Unternehmens durch Einflussnahme auf die Geschäftsprozesse garantieren soll. Beide Ebenen nutzen unterschiedliche Verfahren zur Stärkung der Sicherheit, also zur Kontrolle von Risiken gemäß der Corporate Governance. Die Konzepte beider Ebenen beeinflussen die Ausgestaltung der Sicherheit der Geschäftsprozesse, wie im Folgenden gezeigt.

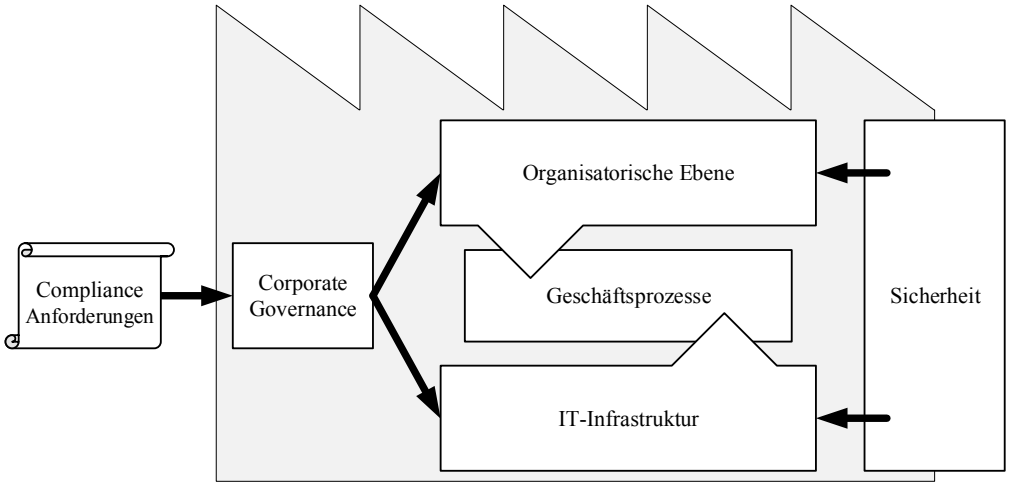


Abbildung 1.2.: Ebenen der Unternehmensarchitektur

1.1.1 IT-Infrastruktur

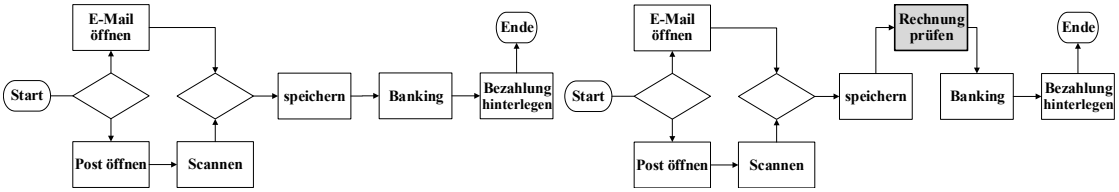
Kern der IT-Infrastruktur ist es, die notwendigen Funktionen für die Ausführung von Geschäftsprozessen bereitszustellen. Auf dieser Ebene befinden sich die be-

reitgestellten Datenelemente, Dienste, Speicher und Kommunikationsmedien. Die Risikokontrolle dieser Ebene folgt den klassischen Schutzzielen der Informationssicherheit (Vertraulichkeit, Integrität, Verfügbarkeit) (Bishop 2004). Zur Erhöhung der Sicherheit zählen Maßnahmen wie Zugriffskontrollen, abhörsichere Datenkanäle, sowie manipulationresistente Speicher zur Protokollierung von Tätigkeiten.

Abbildung 1.3a zeigt einen beispielhaften Rechnungseingangsprozess. Um das Risiko unerlaubten Datenzugriffs zu minimieren, stellt die IT-Infrastruktur in diesem Beispiel durch technische Maßnahmen sicher, dass nur berechtigte Benutzer Zugriff auf die gespeicherten Rechnungen und die Überweisungsfunktionalität ("Banking") erhalten. Zusätzliche Maßnahmen auf Ebene der IT-Infrastruktur können die Integrität und Verfügbarkeit der gespeicherten Rechnungen sicherstellen. Nicht alle Risiken lassen sich jedoch durch technische Maßnahmen wie Zugangskontrolle, Integritätsprüfung oder Redundanz behandeln. Die Sicherheit der IT-Infrastruktur wird daher durch die organisatorische Ebene unterstützt.

1.1.2 Organisatorische Ebene

Abstraktere Anforderungen, wie die geforderte Verlässlichkeit veröffentlichter Finanzdaten, benötigen Eingriffe durch die organisatorische Ebene. Da die technische Sicherstellung der Integrität hinterlegter Rechnungen nicht ausreichend ist um die Rechtmäßigkeit der gespeicherten Daten zu garantieren, bedarf es weiteren Anpassungen, um die noch bestehende Schwachstellen zu behandeln. Hierfür nimmt die organisatorische Ebene Einfluss auf die Geschäftsprozesse, zum Beispiel durch zusätzliche Prüfaktivitäten und gegenseitige Kontrolle der Arbeit durch das Vier-Augen-Prinzip (Botha et al. 2001). Im Beispiel von Abbildung 1.3b ist eine solche Prozessänderung hinsichtlich der Compliance farblich gekennzeichnet. Die Änderungen der Prozessstruktur stellt sicher, dass nur gerechtfertigte Rechnungen bezahlt werden. Zusätzlich fordert die organisatorische Ebene, dass die Überweisung der Beträge durch zwei sich gegenseitig kontrollierenden Personen stattfinden soll – dies stellt eine Kombination organisatorischer und technischen Maßnahmen dar. Erst diese Kombination der Maßnahmen ermöglicht einen sicheren und zur Gesetzgebung konformen Prozess.



(a) Rechnungseingangsprozess

(b) Rechnungseingangsprozess mit Prüfaktivität

Abbildung 1.3.: Beispielprozesse

Maßnahmen der organisatorischen Ebene zur Minimierung eines Sicherheits- oder Non-Compliancerisikos umfassen organisationale, funktionale und informationelle Aspekte (Jablonski et al. 1996).

Organisationale Aspekt

Der organisationale Aspekt umfasst die Ausgestaltung der durch die IT-Infratruktur bereitgestellten Zugangskontrolle (Sandhu et al. 1994). Typischerweise wird die Hierarchie des Unternehmens durch ein Rollenkonzept auf die Zugangskontrolle abgebildet (Ferraiolo et al. 2001). Hierfür bieten sich verschiedene Konzepte an, wie zum Beispiel das Role Based Access Control (RBAC) (Ferraiolo et al. 2001). Die Arbeitsschritte eines Prozesses lassen sich des Rollenkonzeptes entsprechend auf befugte Nutzergruppen zuweisen (Gilbert 1995).

Dieses Konzept ermöglicht es, mit gezielter Rollenzuweisung, Interessenkonflikte durch eine Funktionstrennung oder -bindung (Segregation of Duties (SoD), Binding of Duties (BoD)) zu umgehen (Botha et al. 2001). Diese Prinzipien verhindern, dass ein Mitarbeiter eine Reihe von Aktivitäten ausführen kann, welche einen Betrug begünstigen, indem gewisse, aufeinanderfolgende Tätigkeiten nur durch unterschiedliche Personen durchgeführt werden dürfen. Die Funktionsbindung hingegen fordert die Abarbeitung aufeinanderfolgende Aktivitäten durch die gleiche Person, um die Ausbreitung sensibler Daten zu unterdrücken (Papagiannakopoulou et al. 2015).

Funktionaler Aspekt

Dem funktionalen Aspekt liegen die operationalen Ziele der Prozesse zu Grunde. Dieser Aspekt betrifft den erfolgreichen Ablauf der Geschäftsprozesse entspre-

chend einem dem Unternehmen dienlichen Ziel. Der funktionale Aspekt nimmt Einfluss auf den Kontrollfluss der Prozesse, damit die Tätigkeiten innerhalb des Geschäftsprozesses korrekt (im Sinne der Unternehmensziele) choreographiert sind. Neben dieser grundsätzlichen Anforderung beachtet der funktionale Aspekt kontextabhängige Sicherheitseigenschaften und definiert den Kontrollfluss entsprechend. Bei der Bearbeitung von Kreditanträgen kann dies zum Beispiel bedeuten, dass vor einer Kreditvergabe eine Solvenzprüfung stattfinden muss. Der funktionale Aspekt umfasst auch Konzepte der Nutzungskontrolle (Park et al. 2002), um Vor- und Nachbedingungen zu definieren, welche den Zugriff auf Ressourcen oder die Ausführung von Aktivitäten regeln. Dies ist zum Beispiel die Forderung, dass Vertragsunterlagen innerhalb einer Frist vom Kunden zu unterschreiben sind.

Unternehmen, die personenbezogene Daten verarbeiten, unterliegen dem Datenschutz und müssen die Privatsphäre der Kunden achten. Der funktionale Aspekt dient auch der Einhaltung dieser Anforderungen, indem der Zugriff auf personenbezogene Daten nur Zweckgebunden erfolgen darf, die Speicherzeit begrenzt oder die Anzahl maximaler Zugriffe festlegt wird.

Informationeller Aspekt

Der informationelle Aspekt ist die Perspektive auf die Datenelemente der Prozesse. Am Prozess beteiligte Datenelemente sind häufig entscheidend, welchem Abarbeitungspfad gefolgt wird. Die Höhe eines Kreditantrags bestimmt beispielsweise die zum Vertragsabschluss berechtigten Personen eines Finanzunternehmens.

Eine weitere Sichtweise des informationellen Aspekts ist der Schutz der in den Datenelementen vorhandenen Informationen. Um die Herausgabe unternehmensinterner Daten zu unterbinden, werden bestehenden Zugriffsrechte weiter eingeschränkt. Zur Beantwortung der Frage, ob eine Person Zugriff auf ein Datenelement erhält, ist beispielsweise die Zugriffshistorie von Bedeutung, um ein sogenanntes Chinese-Wall Sicherheitsmodell zu ermöglichen. Dieses Modell unterteilt Informationen in Konfliktklassen und fordert den Verbleib der Information innerhalb der zugeteilten Konfliktklasse. Subjekte, die bereits Zugriff auf sensible Daten eines konkurrierenden Unternehmens hatten, wird der Zugriff untersagt, um die Weitergabe oder Anhäufung von Insiderinformationen zu verhindern. Solche Fälle sind beispielsweise gegeben, wenn eine Beraterfirma konkurrierende Unternehmen betreut. Diese Konzepte der Isolation gelten auch in weiteren Bereichen. Cloud-Dienste, welche

von konkurrierenden Unternehmen genutzt werden, müssen garantieren, dass keine Informationen von einem zu einem anderen Unternehmen fließen kann. Auch innerhalb eines Unternehmens kann die Notwendigkeit der Isolation bestehen, Informationen nur mit den am Prozess beteiligten Personen zu teilen. Dieser Fall erfordert je nach Anforderung zusätzlich die Betrachtung sogenannter verdeckter Kanäle. Sogenannte *Storage-Channels* oder *Timing-Channels* können implizite, verdeckte Informationsflüsse von einer Domäne zu einer anderen ermöglichen (Denning et al. 1977). Die Behandlung dieser Kanäle soll den Informationsgewinn durch Inferenzen unterbinden.

1.1.3 Kombinierte Sicherheit

Die Aspekte der organisatorischen Ebene in Kombination mit der IT-Infrastruktur erlauben sichere Prozesse in Bezug auf folgende Punkte:

- Autorisierung: Die Zugriffskontrolle wird durchgesetzt um sicherzustellen, dass nur autorisierte Personen die am Prozess beteiligten Aktionen durchführen und auf Informationen zugreifen können.
- Nutzungskontrolle: Der Zugriff auf Informationen oder Ressourcen ist nur unter Einbehaltung von Vor- und Nachbedingungen erlaubt.
- Funktionstrennung: Aktivitäten müssen durch unterschiedliche oder mehrere Personen durchgeführt werden, um betrügerisches Handeln zu unterbinden.
- Isolation: Die Trennung in Konfliktklassen und Vermeidung eines Informationsflusses zwischen den Klassen durch direkte oder verdeckte Kanäle.
- Protokollierung: Die revisionssichere Protokollierung von Aktivitäten, um Sicherheitsverletzungen im Nachhinein transparent und somit nachvollziehbar zu machen.

Eine weitere Anforderung an einen sicheren Geschäftsprozess ist die Compliance, welche durch die Kombination der genannten Punkte und entsprechender Gestaltung der Prozessen durch die organisatorische Ebene und dem Geschäftsprozessmanagement (engl. Business Process Management (BPM)) (Becker et al. 2009) erfüllbar sind. Die Umsetzung beider Aspekte erlaubt sichere Prozesse in Bezug auf die IT-Sicherheit und Compliance.

1.2 Einfluss unternehmerischer Interessengruppen

Das Risikomanagement betrachtet die Sicherheit und Compliance der Geschäftsprozesse zum Erhalt des Unternehmens und zur Verfolgung der Unternehmensziele. Diese Ziele sind beeinflusst durch die Stakeholder, den Interessengruppen des Unternehmens (Freeman 2010), wie zum Beispiel, Kunden und Zulieferer, Arbeitnehmer oder die Öffentlichkeit. Diese nehmen direkt und indirekt Einfluss auf die Unternehmenskultur, der Corporate Governance, wie in Abbildung 1.4 angedeutet (Rappaport 1986).

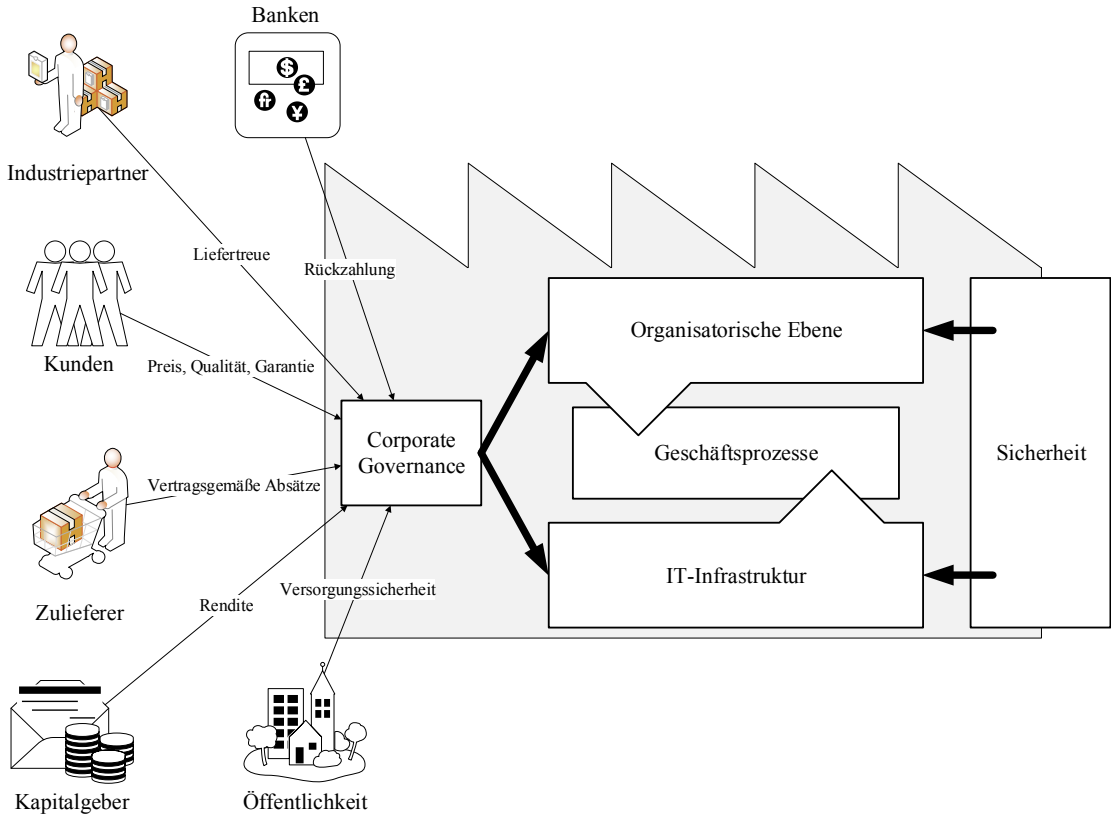


Abbildung 1.4.: Einflussnahme verschiedener Interessengruppen auf die Unternehmensführung.

Entsprechend wird das Unternehmen in weiten Teilen geführt, um Kunden, Kapitalgeber, Zulieferer und Arbeitskräfte zu halten und zu gewinnen (Hubertus 2008) oder ein vorteilhaftes Bild in der Öffentlichkeit zu sichern. Diese verschiedenen Interessen können, wie eingangs erwähnt, die Rentabilität sein – die Kapitalgeber erhoffen sich hohe Renditen. Privat- oder Geschäftskunden verlangen die vertragsgemäße Ausgestaltung der Produkte oder Dienstleistungen zu einem adäquaten

Preis, haben Garantieansprüche und verlangen Zuverlässigkeit und Liefertreue. In einigen Branchen zeigt die Öffentlichkeit ein Interesse gegenüber der Versorgungssicherheit, zum Beispiel bei Energie, dem Gesundheitswesen oder der Abfallwirtschaft.

Die Geschäftsprozesse sind den Stakeholderinteressen dienlich, wenn diese die an sie gestellten Erwartungen erfüllen. Diese Vorgabe wird geprüft, indem Kenngrößen wie Durchlaufzeiten, Ressourcen- und Arbeitsaufwand oder Kosten unter Betrachtung der Unternehmenszielen kontrolliert werden (Krause et al. 2010). Sicherheitsmotivierte Prozessänderungen können negative Auswirkungen auf diese Kenngrößen verursachen. Dies bedeutet, dass die Interessen der Stakeholder nicht mehr entsprechend erfüllt sind, weswegen Stakeholder Maßnahmen zur Erhöhung der Sicherheit ablehnen könnten (Herley 2009) oder übergeordnete Ziele wie der Versorungssicherheit entgegenlaufen.

1.2.1 Automatisierte Geschäftsprozesse

Getrieben durch den globalen Konkurrenzdruck und den Erwartungen der Stakeholder entstanden Produktionsdigmen wie Lean Production (dt.: schlanke Produktion) (Krafcik 1988). Weitere resultierende Bewegungen sind unter dem Schlagwort *Industrie 4.0* zusammengefasst (Spath et al. 2013; Hirsch-Kreinsen 2014). Diese kundenorientierte Rationalisierungsmaßnahmen fördern den Abbau von Überkapazitäten und die Automatisierung der Prozesse (Dombrowski et al. 2015). Auch mittelständische Betriebe bauen auf solche Konzepte gemäß der Leitlinie “das Lager ist die Straße” (Pfluger 2005). Mit diesem Leitbild prägte der Vorstand Manfred Bene des gleichnamigen, mittelständischen Möbelherstellers die Modernisierung seines Unternehmens hin zu Just-in-Time-Lieferketten. Um diesem Konzept zu folgen, sind zeitlich aufeinander abgestimmte Prozesse gefordert, damit Waren oder Produkte gerade dann eintreffen, wenn sie gebraucht werden.

Dieser Trend, ermöglicht durch automatisierte Geschäftsprozesse, setzt sich bei Unternehmen unterschiedlicher Größe durch und wird durch vielfältige, staatliche Projekte wie z.B. GESINE (Eymann et al. 2014; BMWi 2014) vorangetrieben. Die Projekte zeigen die Einsatzfähigkeit automatisierter Geschäftsprozesse, um die Arbeitsabläufe effizienter zu gestalten und Kosten einzusparen.

Mit diesen Trends steigen allerdings die Dependenzen der Prozesse analog zur Produktivität. Die von automatisierten Geschäftsprozessen stramm choreographierte Nutzung von Ressourcen lässt sich bereits durch kleine Änderungen empfindlich stören. Die von Perrow[2] benannte Normal Accident Theory (NAT) zeigt auf, dass in komplexen, aufeinander aufbauenden Systemen Störungen nicht vermeidbar sind und der Wegfall von selbst als unwichtig empfundenen Komponenten die Unternehmensabläufe so behindern können, dass ganze Geschäftsprozesse auf Grund des Peitscheneffekts (Lee et al. 1997) gestört und nicht mehr verfügbar sind (Perrow 1994). Deren Ausfall wirkt sich weiter auf das Zusammenspiel der Prozesse – der Prozessarchitektur aus. Diese ist im Falle der schlanken Produktion oder automatisierter Prozesse jedoch gerade darauf optimiert, effizient zu arbeiten und weist daher wenig Redundanzen oder Puffer auf (Dickmann 2008).

1.2.2 Spannungsfeld Verfügbarkeit, Sicherheit und Compliance

Die Sicherheit und Compliance ist nicht die alleinige Zielsetzung des Risikomanagements. Stattdessen werden jegliche unternehmerische Risiken, die zu disruptiven Effekten führen können betrachtet, da wie beschrieben, gerade diese Effekte große Auswirkungen auf die automatisierten Prozesse zeigen. Der Ausfall der Produktion, Lieferuntreue oder Störungen der Geschäftsprozesse sind damit weitere Elemente der Betrachtung des Risikomanagements (Diederichs 2013). Studien über die Verluste durch nicht fristgerechte Vertragseinhaltungen oder Dienstleistungen zeigen einen durchschnittlichen Verfall der Firmenwerte um 40 % (Hendricks et al. 2005). Um die Risiken eines solchen Verlusts durch zeitlich verzögerte Prozesse zu behandeln, bedarf es der Betrachtung der Wechselwirkungen eingeführter Sicherheits- und Compliancemaßnahmen auf die Abarbeitung der Geschäftsprozesse.

Aus Sicht der Stakeholder ist der “Return-of-Investement”, die Kapitalrendite, eines der wichtigsten Kenngrößen einer Maßnahme. Dies gilt ebenso für Sicherheits- oder Compliancemaßnahmen. Laut einer Umfrage mit 4000 Unternehmen aus 28 Ländern werden ungefähr 1/5 des IT-Budgets für die IT-Sicherheit ausgegeben. Dies entspricht im Mittel dem 2,5-fachen der verursachten Folgekosten eines erfolgreichen Angriffs oder Betrugsfalls (Kaspersky Lab 2016). Zu einem Großteil entstehen diese Folgekosten durch verpasste Gelegenheiten und Geschäftsabschlüsse. Dies lässt sich auf gestörte Geschäftsprozesse zurückführen, welche auf Grund

[2] Vgl. Perrow 1994.

des erfolgten Vorfalls nicht mehr fristgerecht abgearbeitet werden konnten. Unternehmensführungen, von kleinen und mittelständischen Betrieben als auch von Konzernen erkennen laut dem Bericht, dass die IT-Sicherheit nicht alleinig durch sichere Hard- und Software realisierbar ist, sondern auch durch entsprechend gestaltete Geschäftsprozesse. Maßnahmen zur Erhöhung der Sicherheit können ebenso die Umgestaltung von Geschäftsprozessen bedeuten. Der Bericht fasst die Wichtigkeit der Kostenbetrachtung der Sicherheit zusammen. Nur mit einer kosteneffizienten Umsetzung können Unternehmen die steigende Zahl der Angriffe auf IT-Infrastruktur mit den gegebenen finanziellen Mitteln abwehren. Dies bedeutet auch, dass die Betrachtung der Auswirkungen von Sicherheitsmaßnahmen auf die Termintreue laufender Geschäftsprozesse von großer Bedeutung ist.

Unter Betrachtung von Industrie 4.0 und verbundenen Trends wie Lean Production oder Just-In-Time-Lieferketten, ist die bedeutende Voraussetzung zur Erfüllung der Stakeholderinteressen die effiziente, zeitlich determinierte, also fristgerechte Ausführung der Geschäftsprozesse (Hubertus 2008). Verzögerungen erhöhen die Kosten, können Strafzahlungen hervorrufen und wirken sich auf abhängige Prozesse aus. Die Verfügbarkeit eines Geschäftsprozesses aus Sicht der Stakeholder ist somit die Anforderung, dass dieser fristgerecht zu einem Ausgang führt. Maßnahmen der Risikominimierung, wie die Einführung von Sicherheits- oder Complianceumsetzungen dürfen diese Verfügbarkeit der Prozesse nicht gefährden.

Das vorab, in Abschnitt 1.1.2 gezeigte Beispiel sicherheitsorientierter Geschäftsprozessänderungen aus Abbildung 1.3 zeigt exemplarisch auf, wie sich die Sicherheitsumsetzungen durch die IT-Infrastruktur und der organisatorischen Ebene auf einen Prozess auswirken können. Zwar verringern die gezeigten Maßnahmen das Risiko eines Betrugs, doch zeigen die Änderungen nachteilige Effekte. So kann eine Zugriffskontrolle die Ausführung eines Geschäftsprozesses blockieren, wenn autorisierte Personen ausfallen (Holderer et al. 2016). Zusätzliche, der Sicherheit dienliche Aktivitäten erhöhen den Arbeitsaufwand und führen, wie auch eine gegenseitige Kontrolle mehrerer Mitarbeiter, zu einem erhöhten Ressourcen- und Zeitbedarf. Für die Akzeptanz einer Maßnahme zur Erhöhung der Sicherheit oder der Minimierung eines Risiko ist es unabdingbar, die dadurch entstehenden Kosten mit dem erwünschten Effekt zu vergleichen. Sinnhaft ist eine solche Maßnahme aus Sicht der Stakeholder oder Unternehmensführung, wenn die erwartete Risikominimierung höher ausfällt als die hierdurch entstehenden Nachteile. Die Determinierung

dieser Kosten ist von Unsicherheiten geprägt und benötigt eine Sicht auf die Unternehmungsprozesse und deren Zusammenwirken (Mercuri 2003).

Um in diesem Spannungsfeld zwischen Sicherheit und Compliance auf der einen Seite und den Stakeholder- und Unternehmensinteressen auf der anderen Seite eine Lösung zu finden, welche durch die Stakeholder Unterstützung findet und den Unternehmenszielen nicht entgegen läuft, müssen die Auswirkungen der Maßnahmen in Hinblick auf die Verfügbarkeit und der Kosten erkennbar sein. Hierdurch können Prozessverantwortliche Lösungen zur Umsetzung mit minimalen, negativen Auswirkungen auf die Verfügbarkeit finden.

1.3 Beitrag der Arbeit

Diese Arbeit betrachtet die Fragestellung, wie eine Abschätzung der Auswirkungen durch Änderungen bestehender Prozesse auf eine bestehende Prozessarchitektur möglich ist. Auch Prozessumsetzungen, die der Sicherheit von Geschäftsprozessen dienliche sind, dürfen die Produktivität des Unternehmens nicht gefährden. Trends wie Industrie 4.0 (Spath et al. 2013) oder Lean-Production (Krafcik 1988) führen zu zeitlich aufeinander abgestimmten Prozessen mit geringeren Lagerbeständen und hohen Abhängigkeiten (Feltham et al. 2013; Morin et al. 2001). Auf Grund der entstehenden Interdependenzen wirkt sich die Änderung eines Prozesses nicht mehr lokal aus, sondern auch auf abhängige Prozesse. Die Betrachtung dieser Abhängigkeiten ist notwendig, um die Auswirkungen allgemeiner oder sicherheitsrelevanter Anpassungen der Prozesse durch die organisatorische Ebene oder der IT-Infrastruktur zu quantifizieren. Das in dieser Arbeit entwickelte Geschäftsprozesssimulationsverfahren soll Unternehmen dazu befähigen, verschiedene Umsetzungen ihrer Sicherheitsanforderungen bereits während der Prozessentwicklung im Hinblick auf das Zeitverhalten, den Ressourcenverbrauch sowie die Kosten miteinander zu vergleichen, damit die Sicherheit von Prozessen nicht die Interessen der Stakeholder vereitelt.

Die Simulation ermöglicht zusätzlich, Prozesse während ihrer Ausführung so zu koordinieren, dass Interdependenzen vermieden werden, um Ressourcenengpässen oder Personalkonflikten auszuweichen und Wartezeiten zu verkürzen. Dies ist nur mit Kenntnissen über die Ressourcenlage und des Zeitverhaltens der Aktivitäten

möglich. In dieser Arbeit werden Verfahren vorgestellt, um diese Informationen aus bestehenden Daten zu extrahieren und nutzbar zu machen.

Zusammengefasst ergeben sich die folgenden Forschungsfragen:

Wie lassen sich Auswirkungen einer sicherheitsdienlichen Änderung eines Prozesses auf abhängige Prozesse bestimmen?

Sind die Abhängigkeiten zwischen den Prozessen bekannt, kann eine gezielte Steuerung der Einzelaktivitäten erfolgen, um die Effekte der Abhängigkeiten zu verringern. Die zweite Forschungsfrage lautet daher:

Ist eine Steuerung der Aktivitäten möglich, um eine gegenseitige Beeinflussung der Prozesse zu vermeiden?

Zur Beantwortung der Fragen sind Annahmen über die Abhängigkeiten der Prozesse notwendig. Die letzte Forschungsfrage adressiert diese Problemstellung:

Wie sind realistische Annahmen über die Abhängigkeiten der Prozesse möglich?

Die Beantwortung der Forschungsfragen erlaubt es Prozessverantwortlichen die Geschäftsprozesse so zu gestalten, dass sie einerseits die notwendigen Anforderungen an Sicherheit und Compliance erfüllen und anderseits die Unternehmenszielen gemäß der Corperate Governance verfolgen.

1.4 Anwendungsfall

Im Rahmen der Förderinitiative eStandards (BMWi 2014) des Bundesministeriums für Wirtschaft und Energie entstand das Projekts GESINE mit dem Ziel kleinen und mittelständische Unternehmen den Nutzen automatisierter Geschäftsprozesse aufzuzeigen und deren Einsatz zu steigern (Eymann et al. 2014). Hierfür wurden Industriepartner angeworben, deren Prozesse (teil-)automatisiert werden konnten. Prozesse der Buchhaltung, wie die Rechnungsstellung, zeigten große Potenziale zur Automatisierung durch den elektronische Rechnungsabwicklungsstandard ZUGFeRD (AWV e.V. 2014; Elter 2014). Bereits kleinste Betriebe nutzen

digitale Rechnungen und wollen diese verstärkt einsetzen (Biffar 2016). Die elektronische Rechnung resultiert in schnelleren Prozessen mit geringerem Personalaufwand und Kosten. Das Projekt GESINE zeigte hiermit die Fähigkeit, auch die Prozesse kleinerer Unternehmen zu automatisieren. Die Kooperation mit einem süddeutschen Elektronikunternehmen zeigte den erfolgreichen Einsatz zentral gesteuerter, digitaler Geschäftsprozesse. Das Unternehmen nutzte Software der SAP AG für ihre Bestellprozesse. Die Betrachtung der Aktivitätsprotokolle der Prozesssteuerungssoftware zeigte, dass trotz teils manueller Bearbeitung in über 60 % der Fälle die untersuchten Prozesse strikt dem definierten Prozessablauf folgten.

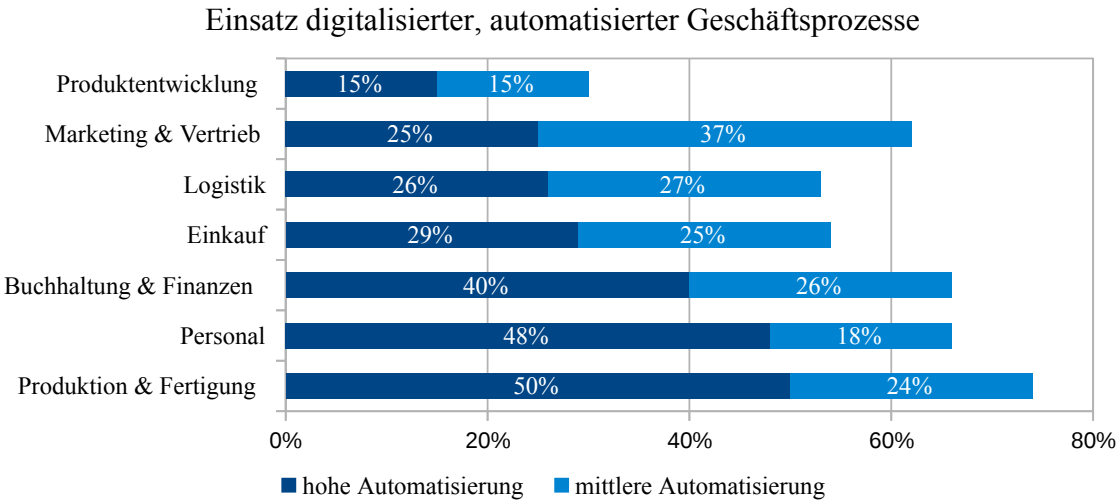


Abbildung 1.5.: Statistik automatisierter Geschäftsprozesse[3].

Die in dieser Arbeit entworfenen Methoden gelten für digital unterstützte, automatisierte und aus diesem Grund exakt spezifizierten Prozesse. Wie eine durch die Bitkom durchgeführte Studie zeigte, werden solche Prozesse mittlerweile in vielen Bereichen eingesetzt und weisen großes Potenzial für weitere Einsatzgebiete, z.B. im Bereich der Industrie 4.0 auf (Biffar 2016; Spath et al. 2013), siehe Abb. 1.5. Derartige Geschäftsprozesse bilden den Anwendungsfall dieser Arbeit.

1.5 Struktur der vorliegenden Dissertation

In Kapitel 2 werden zunächst die Interdependenzen von Prozessen innerhalb einer Prozessarchitektur betrachtet. Es folgt ein Überblick über das Themenfeld des

[3]Vgl. (Biffar 2016).

Geschäftsprozessmanagements und dessen Methoden zur Steuerung, Anpassung und zur Prüfung der Prozessleistungen. Das Kapitel schließt mit der Spezifikation notwendiger Modellierungsfunktionen eines sicherheits- und complianceorientierten Simulationsverfahrens von gegenseitig abhängigen Prozesse ab.

Hierfür werden in Kapitel 3 die für eine Simulation von Geschäftsprozessen anwendbaren Ansätze und deren Modelle genauer dargestellt und mit den gestellten Anforderungen verglichen.

In Kapitel 4 wird mit den Ressource-Timed Petri Nets (RTPN) eine neue, in dieser Arbeit entwickelte Erweiterung bestehender Simulationsmodelle präsentiert, da keines der untersuchten Modelle die gestellten Anforderungen erfüllt. Das Kapitel schließt mit der Frage ab, wie sich ein RTPN-Modell mit realistischen Daten erstellen lässt. Zu diesem Zweck werden im Rahmen der Arbeit entwickelte Methoden vorgestellt, um diese aus Ausführungsprotokollen der IT-Systeme zu gewinnen.

Die Evaluation des RTPN-Simulationsverfahrens in Kapitel 5 demonstriert die Tauglichkeit des Ansatzes. Es wird weiterhin gezeigt, wie sich die verwendeten Methoden nutzen lassen, um Geschäftsprozesse derart zu steuern, dass Prozesse möglichst ohne Störungen innerhalb einer Prozessarchitektur ablaufen.

Mit der Zusammenfassung in Kapitel 6 schließt die Dissertationsschrift ab.

Kapitel 2

Interdependenzen sicherer Geschäftsprozesse

Die in der Einleitung erwähnte Problemstellung Geschäftsprozesse sicher und konform zur Gesetzgebung umzusetzen und gleichzeitig die Stakeholderinteressen zu vertreten, erfordert die Betrachtung der Interdependenzen der Geschäftsprozesse. Ohne eine solche Betrachtung ist es nicht möglich, die Effekte sicherheitsbezogener Anpassungen ganzheitlich zu erfassen, da sich diese über Prozessgrenzen hinweg propagieren.

Zu Beginn des Kapitels werden die Interdependenzen von Geschäftsprozessen betrachtet. Daraufhin wird in diesem Kapitel beschrieben, mit welchen Methoden Unternehmen ihre Prozesse erstellen und steuern. Mit diesen Verfahren setzen Unternehmen die Sicherheit in ihren Geschäftsprozessen um. Es wird gezeigt, dass diese Umsetzungen zwar einer Kontrolle der Prozesskosten unterzogen werden, aber die Auswirkungen von sicherheitsbezogenen Änderungen an Prozessen nicht ausreichend vorhersehbar sind, da die Interdependenzen der Geschäftsprozesse keine Berücksichtigungen finden. Abschließend werden Kriterien für eine sicherheitsorientierte Simulation der Interdependenzen in Geschäftsprozessen definiert.

2.1 Ressourcendimension der Geschäftsprozesse

Die Digitalisierung der Geschäftsprozesse brachte zentrale Instanzen zur Steuerung, Kontrolle und Überwachung von Geschäftsprozessen hervor. Diese sogenann-

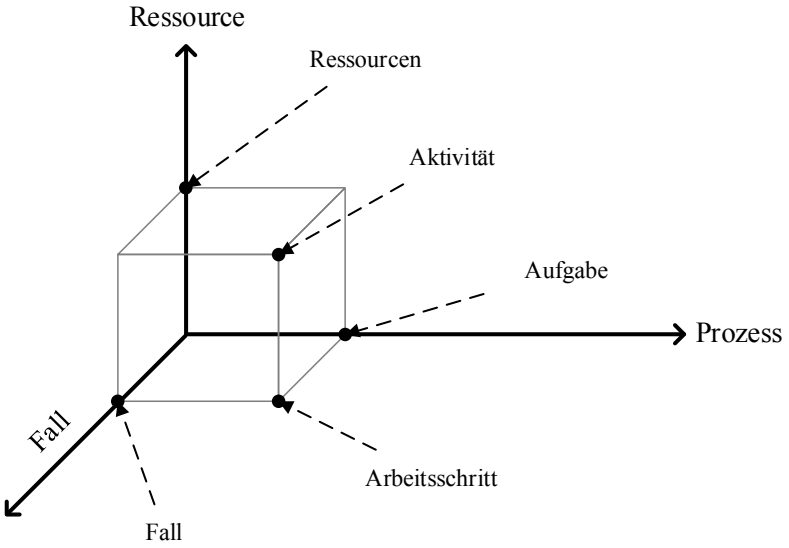


Abbildung 2.1.: Dimensionen eines Workflows[1]

ten Workflow Management System (WfMS) steuern und regeln die Abarbeitung von Aktivitäten (Aalst et al. 2004; Koulopoulos 1995). In "The application of Petri nets to workflow management" identifiziert Aalst drei Dimensionen eines Prozesses, welche durch ein WfMS gesteuert werden (Aalst 1998a):

- Die Fall-Dimension
- Die Prozess-Dimension
- Die Ressourcen-Dimension

Die Abarbeitung eines Prozesses unterliegt entsprechend den Dimensionen einem Fall und den dafür eingesetzten Ressourcen, wie in Abbildung 2.1 gezeigt.

Die Fall- und Prozess-Dimension stellt eine Sicht auf die Prozess-Instanzen, beziehungsweise auf den Prozess als solchen dar. Die Ressourcen-Dimension hat Einfluss darauf, wie ein geplanter Prozess auf der entsprechenden Dimension sich im einzelnen Fall, der konkreten Situation, verhält. Eine einzelne Aktivität unterliegt allen drei Dimensionen und hängt vom zu bearbeitenden Fall, dem Prozess und den verwendeten Ressourcen ab. Um Unternehmensziele zu erreichen, muss ein Unternehmen Einfluss auf alle Achsen ausüben. Die Ressourcen-Dimension

[1] nach Aalst 1998a.

ist die einzige Achse, die im Gegensatz zu den anderen Achsen von Unternehmen nicht vollständig kontrollierbar ist. Prozesse und Prozess-Instanzen lassen sich entwickeln, verbessern und instantiieren. Zwar können Ressourcen geschaffen und geplant werden, doch ist ein Ausfall von Ressourcen nicht vorhersehbar oder sie liegen wegen Produktions-Digmen wie Lean Production (Brunner 2014) häufig nicht mehr in der eigenen Verantwortung. Zusätzlich werden Ressourcen von anderen Prozessinstanzen verwendet und deswegen für weitere Prozesse blockiert. Einzelne Aktivitäten sind dann nicht fristgerecht verfügbar – sie müssen auf die Freigabe der von ihnen benötigten Ressourcen warten. Änderungen an Prozessen durch Compliance- oder Sicherheitsmaßnahmen können die Anzahl blockierter Ressourcen oder die Ressourcenabhängigkeit weiter erhöhen, wenn diese zum Beispiel mehr Personal oder Ressourcen binden. Wegen neuen Entwicklungen wie Just-in-Time-Lieferketten reagieren Prozesse und Aktivitäten stärker auf die Ressourcendimension, da Lagerbestände nur noch minimal sind und keine Puffer, Überkapazitäten oder Redundanzen aufweisen (Wildemann 2001).

Ressourcen zeigen unterschiedliche Ausprägungen. Sie können explizit sein, also jeweils nur einmal gleichzeitig genutzt werden, oder sind geteilt, also parallel von mehreren Aktivitäten gleichzeitig nutzbar. Ressourcen zeigen damit ein unterschiedliches Verhalten und es muss abgewägt werden, welche Ressourcen welchen Prozessen zugeteilt werden, damit die Prozesse innerhalb der Prozessarchitektur effizient und ohne Verzögerungen arbeiten.

Die Ressourcen-Dimension stellt damit eine Interdependenz der Prozessen dar. Prozesse teilen sich Ressourcen und sind gegenseitig davon abhängig, diese freizugeben. Sind Ressourcen blockiert, kann die Abarbeitung der Prozesse nicht mehr gemäß den Anforderungen erfüllt werden und die Prozesse stehen dem Unternehmen nicht mehr gemäß den auferlegten Zielen des Corporate Governance und der Stakeholder zur Verfügung. Für die Betrachtung der Auswirkungen von Sicherheit und Compliance auf die Verfügbarkeit der Geschäftsprozesse ist es somit notwendig, die Ressourcen-Dimension der Prozesse zu betrachten, um Stakeholderinteressen zu wahren.

2.2 Geschäftsprozessmanagement

Verantwortlich für die Umsetzung der Sicherheit und Compliance in Geschäftsprozessen ist das BPM. Das Gebiet des BPM, oder Geschäftsprozessmanagement (GPM) im Deutschen, umfasst die Entwicklung, Verbesserung, Steuerung und Dokumentation von Geschäftsprozessen (Weske 2012; Becker et al. 2009). Mit dieser Prozessperspektive nehmen Unternehmen Einfluss auf die gezeigten Dimensionen ihrer Prozesse. BPM beschreibt hierfür einen standardisierten Ablauf (Aalst et al. 2003). Dieser Ablauf wird durch den BPM-Lebenszyklus beschrieben. Er umfasst die Phasen Entwurf, Konfiguration, Ausführung und Überwachung, wie in Abbildung 2.2 veranschaulicht. Im Falle bereits bestehender Prozesse beginnt der Lebenszyklus statt mit dem Entwurf von Geschäftsprozessen, mit der Optimierung dieser. Der Einstiegspunkt des BPM-Lebenszyklus ist in der Abbildung 2.2 grau hinterlegt. Mit BPM zielen die Unternehmen auch auf die Standardisierung und Automatisierung ihrer Geschäftsprozesse, um sich Wettbewerbsvorteile zu verschaffen (Best et al. 2010).

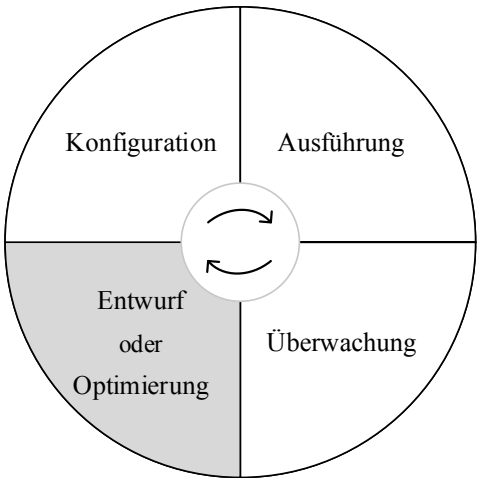


Abbildung 2.2.: Prozessmanagement Lebenszyklus[2]

Die Bedeutungen der einzelnen Schritte sind:

- Entwurf: In dieser Phase werden die bestehenden, nicht digitalisierten Prozesse aufgenommen und in eine maschinenlesbare Sprache überführt. BPM nutzt hierfür standardisierte Notationsformate, um die Prozesse zu modellieren und

[2]Siehe Aalst et al. 2003

maschinenlesbar zu gestalten. Das am weitesten verbreitete Modellierungsformat ist das Business Process Model and Notation (BPMN) Format. Diese Prozess-Sprache ist von vielen Modellierungswerkzeugen und automatisierten Prozess-Systemen unterstützt (Grosskopf et al. 2009).

- Konfiguration: Die in BPMN überführten Prozesse können durch WfMS automatisiert und ausgeführt werden. Jedoch ist es notwendig, das WfMS in dieser Phase zuerst zu konfigurieren, Benutzer und Rollen zuzweisen sowie die einzelnen Aufgaben zu spezifizieren.

- Ausführung: In dieser Phase wird der alte, bestehende Prozess nicht weiter ausgeführt und stattdessen der entsprechende digitalisierte Prozess durch das WfMS umgesetzt. Die Durchführung, das heißt die Instantiierung einzelner Instanzen auf Grundlage des Prozessessmodells geschieht von nun an digital unterstützt. Einzelne Aktivitäten sind entweder vollständig innerhalb des WfMS abgebildet oder das System protokolliert die Durchführung der Aktivitäten.

- Überwachung: Während der Abarbeitung der Prozesses protokolliert das WfMS in sogenannten Prozess-Logs Informationen über den Prozessablauf. Das WfMS speichert typischerweise zu jeder Aktivität den Zeitpunkt der Ausführung, von welchem Benutzer diese bearbeitet wurde und welche Betriebsmittel dafür genutzt wurden. Je nach Ausprägung enthalten diese Protokollierungen weitere Informationen, zum Beispiel aktivitätsspezifische Daten. Die in dieser Phase stattfindende Überwachung dieser Daten gibt Aufschluss darüber, ob der Prozess wie gefordert funktioniert.

- Optimierung: Statt Prozesse zu digitalisieren, kann der BPM-Lebenszyklus auch mit der Verbesserung bestehender Prozesse beginnen. In dieser Phase findet eine tiefgreifende, teils automatisierte Betrachtung der Prozess-Logs statt, um Hinweise auf Engpässe oder Ereignisse zu bekommen, welche den Prozessablauf behindern. Dieses Vorgehen ist ein Teilgebiet des Process Mining (PM) und hilft dabei, bestehende Prozesse zu optimieren oder neu zu entwickeln.

BPM stellt ein Instrument dar, Prozesse zu digitalisieren und zu standardisieren. Dies dient nicht dem Selbstzweck, sondern bietet Möglichkeiten, diese besser zu Überwachen und an aussagekräftige Informationen zu gelangen, um wiederum

Verbesserungen vorzunehmen. Diese Vorgehensweise ermöglicht es, Änderungen an Prozessen oder Aktivitäten durchzuführen, um Compliance-Richtlinien umzusetzen, die Sicherheit zu erhöhen oder um die Prozesse auf sich verändernde, äußere Umstände anzupassen.

2.2.1 Process Mining zur Ermittlung des Prozessverhaltens

Damit die Geschäftsprozesse entsprechend den in der Entwurfsphase verfolgten Absichten ablaufen, ist die Kontrolle und Überwachung der laufenden Prozesse notwendig. Innerhalb des Geschäftsprozessmanagements leistet das Teilgebiet des Process-Minings (PM) hierfür Unterstützungen in weiten Bereichen des BPM-Lebenszyklus. Die Techniken von PM arbeiten auf Log-Dateien, genauer auf Prozess-Logs. Dies sind durch die WfMS oder andere Instanzen maschinell erstellte Protokolle über die Abarbeitung von Aktivitäten. Einzelne Einträge in einem Prozess-Log entsprechen den Aktivitäten eines Geschäftsprozesses und beinhalten weitere Informationen, wie den Bearbeiter der Aktivität, die Zeitdauer der Bearbeitung und eingesetzte Ressourcen oder angehängte Daten. Die erzielbaren Ergebnisse und Resultate von PM sind vielfältig und lassen sich folgendermaßen unterteilen (Aalst 2016; Aalst et al. 2011):

- PM unterstützt die Aufdeckung oder Rekonstruktion bestehender, jedoch noch nicht erfasster Prozesse und unterstützt somit die Entwurfsphase des BPM-Lebenszyklus (Van Dongen et al. 2009). Die Aufdeckung von Prozessen arbeitet auf den Prozess-Logs und liefert ein Prozessmodell, welches den Aktivitätsausführungen des Prozess-Logs entspricht.

- PM ermöglicht die Prüfung der korrekten Ausführung von Prozessen (Cook et al. 1999). Ausgehend von Prozessmodellen und Prozess-Logs liefert die Prüfung Aussagen, welche Ausführungen nicht konform mit dem Modell waren. Dieses Vorgehen findet Anwendung, um die Einhaltung von Prozessvorschriften zu prüfen, z.B. die Umsetzung des Vier-Augen-Prinzips.

- PM fördert die Verbesserung bestehender Prozesse (Aalst 2012). Die kombinierte Nutzung von Prozess-Logs und Modellen ermöglicht eine Identifizierung von prozessspezifischen Merkmalen mit negativen Einflüssen auf die Leistungskennzahlen der Prozesse. Dies können beispielsweise organisationelle Zusammenhänge sein (verschiedene Abteilungen zeigen unterschiedliche

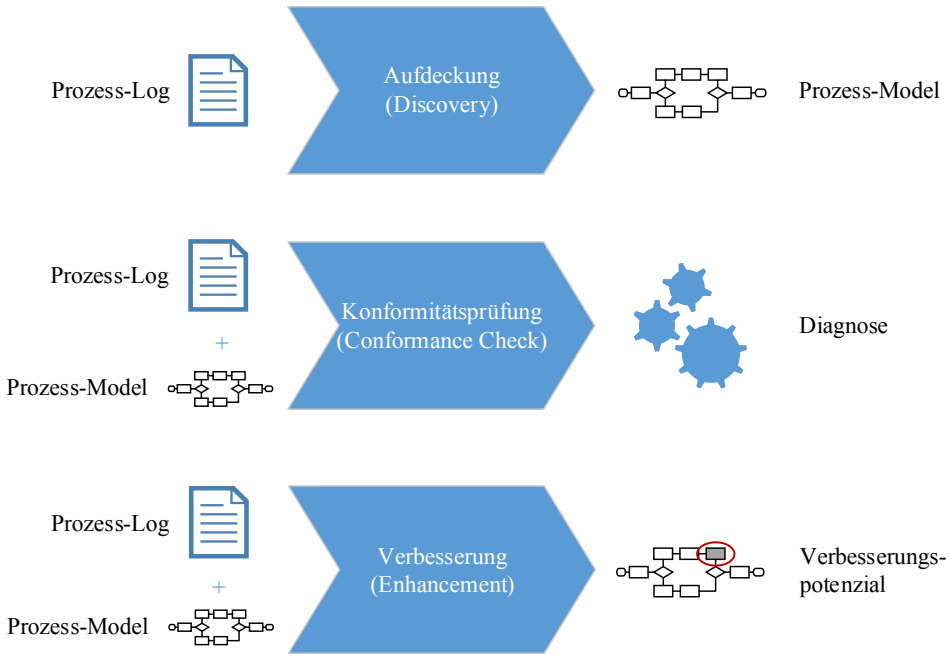


Abbildung 2.3.: Aspekte von Process Mining

Zeitverhalten bei der Abarbeitung des Prozesses), die Erkennung zu langer Prozesspfade oder zu ineffizienter Ressourcennutzung.

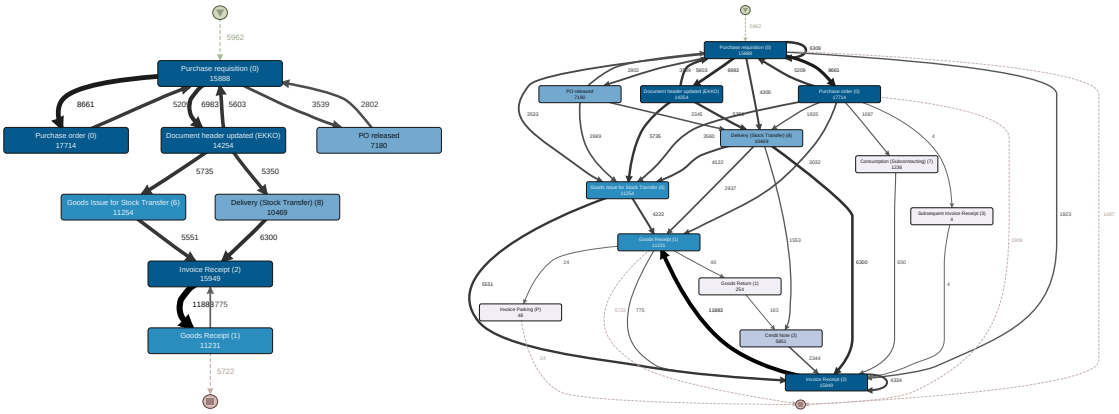
Diese drei Teilaspekte von PM, die verwendeten Modelle und ihre Ausgaben sind in Abbildung 2.3 grafisch aufgearbeitet.

Rekonstruktion von Prozessmodellen

Die Rekonstruktion von Prozessmodellen (Process Discovery) ist dienlich, wenn in einem Unternehmen maschinell unterstützte Prozesse durchgeführt werden, für die kein Prozessmodell existiert oder das Prozessmodell nicht mehr der Realität entspricht (Van Dongen et al. 2009). Hiermit unterstützt das PM die Entwurfsphase des BPM-Lebenszyklus. Ausgehend von den protokollierten Aktivitäten, nutzt die Rekonstruktion kausale Zusammenhänge der im Prozess-Log gefundenen Abarbeitungsreihenfolge, um den Kontrollfluss des Prozesses nachvollziehen. Abhängig von dem Verwendungszweck können strukturbasierte Verfahren der Prozessrekonstruktion, die einzelnen Prozessausführungsschritte in ein präzises Prozessmodell überführen.

Präzise Prozessmodelle bilden zwar jede Ausführungsmöglichkeit ab, die im Prozess-Log aufgetreten ist, sind aber wegen ihrer Komplexität ungeeignet, das allgemeine Prozessverhalten zu erkennen. Heuristische Verfahren, oder *Fuzzy-Mining* (Günther et al. 2007) betrachten die häufigsten Ausführungsschritte, um ein abstrakteres, jedoch leicht verständliches Prozessmodell zu rekonstruieren. Die erzeugten Modelle bilden zwar nicht jeden der gefundenen Ausführungsschritte ab, sind jedoch einfacher verständlich.

Ein Beispiel unterschiedlich abstrakt rekonstruierter Prozessmodelle ist in Abbildung 2.4 gezeigt. Die Varianten zeigen ein rekonstruiertes Prozessmodell des gleichen Bestellprozesses aus Prozess-Logs eines süddeutschen Unternehmens. Beide Modelle nutzen das gleiche Prozess-Log, abstrahieren jedoch unterschiedlich. Das Modell in Abb. 2.4a zeigt eine hohe Abstraktion. Hier wurden nur die häufigsten Aktivitäten und deren Abarbeitungsreihenfolge in das Prozessmodell überführt. Abb. 2.4b zeigt mehr Aktivitäten und Verbindungen zwischen den Aktivitäten. Damit deckt sich das Modell genauer mit dem Prozess-Log, ist jedoch dem allgemeinem Prozessverständnis weniger dienlich.



(a) Rekonstruiertes Prozessmodell mit hoher Abstraktion

(b) Rekonstruiertes Prozessmodell mit geringerer Abstraktion

Abbildung 2.4.: Aus einem Prozess-Log rekonstruierte Prozessmodelle unterschiedlichem Abstraktionsgrades[3]

Konformatitätsprüfung

Die zuvor gezeigten Prozessmodelle in Abbildung 2.4 zeigen einen Prozess auf Basis von Prozess-Logs. Das zu Grunde liegende Prozessmodell des Unternehmens

[3]Erstellt mit Disco (Günther et al. 2012)

ist in Abbildung 2.5 gezeigt. Die durch die Rekonstruktion entstandenen Modelle zeigen im Vergleich zu dem geplanten Prozessmodell Diskrepanzen. Diese Diskrepanzen entstehen auf Grund verschiedener Umstände, wie zum Beispiel nachträglicher Veränderungen der Bestellung, falscher Lieferungen, Teilbezahlungen oder aus Krankheitsgründen, wenn der Prozess durch eine vertretende Personen beendet werden muss, die nicht mit dem Prozess vertraut ist.

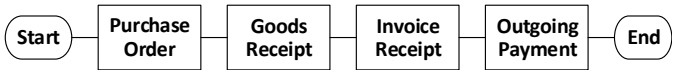


Abbildung 2.5.: Prozessmodell eines Bestellprozesses

Diese entstandenen Prozessvariationen können vorsätzlich oder unabsichtlich zu sicherheitskritischen Situationen führen, wenn Aktivitäten übersprungen oder Zugangskontrollen umgangen werden. Auch ein absichtlicher Betrugsfall oder ein Verstoß gegen gesetzliche Auflagen kann zu einer Prozessvariation führen, wenn angenommen wird, dass das zu Grunde liegende Prozessmodell sicher und gesetzeskonform ausgestaltet ist. Die Konformitätsprüfung umfasst diese Prüfung der Prozess-Logs auf Konformität mit dem Prozessmodell (Accorsi et al. 2012). Hierfür wird verglichen, ob die in einem Prozess-Log extrahierten Aktivitätsreihenfolgen dem Prozessmodell entsprechen. Mit Modellprüfverfahren (Model-Checker) ist es möglich, logische Zusammenhänge der Prozess-Logs zu prüfen (Zahoransky et al. 2016; Accorsi et al. 2014). So lässt sich sicherstellen, dass zum Beispiel stets eine Aktivität B auf eine Aktivität A folgte, Zugangskontrollen eingehalten wurden oder gewisse Aktivitäten nicht in Kombination auftraten. Die Konformitätsprüfung unterstützt hierdurch die Überwachungsphase des BPM-Lebenszyklus.

Diese Prüfverfahren stellen neben der sicheren Ausgestaltung der Geschäftsprozesse ein zusätzliches Merkmal der Sicherheit in Unternehmen dar. Diese nachträgliche Konformitätsprüfung erlaubt Prozessabweichungen, die zu Betrugsfällen, Sicherheitsverletzungen oder der Nichteinhaltung von Complianceauflagen führten, im Nachhinein aufzudecken (Lima Bezerra et al. 2009). Somit lassen sich die ausgenutzten Sicherheitsschwachstellen erkennen und die verantwortlichen Personen eines erfolgten Betrugs identifizieren.

Unterstützung des PM zu Prozessverbesserungen

Um Geschäftsprozesse mit Verbesserungspotenzial identifizieren zu können, sind Prozessentwickler auf die Überprüfung der Prozesse gemäß verschiedener Leistungsangaben angewiesen. Die Sicht auf Prozess-Logs liefert in diesem Teilgebiet beispielsweise Informationen über das Zeit- oder Ressourcenverhalten der Aktivitäten innerhalb von Prozessen, um Engpässe oder Aktivitäten mit verzögerter Abarbeitung grafisch zu erkennen, wie in Abbildung 2.6 dargestellt. Die Abarbeitung der Aktivitäten wird in diesem Beispiel als Marken dargestellt, welche durch das Prozessmodell traversieren.

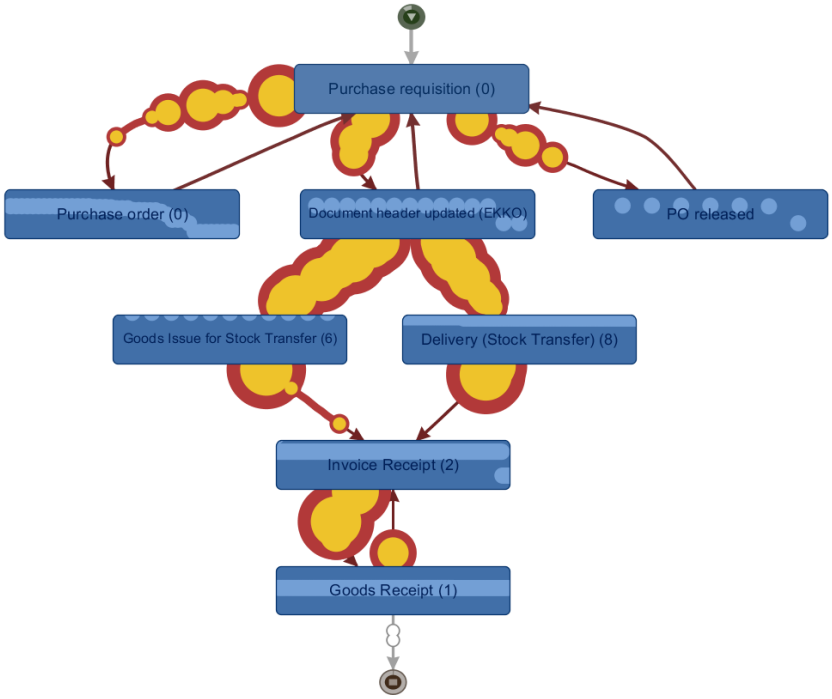


Abbildung 2.6.: Visualisierung des zeitlichen Verhaltens der Abarbeitung von Aktivitäten an Hand bewegender Marken im rekonstruierten Prozessmodell[4]

Die extrahierten Daten lassen sich zusätzlich filtern, um interessante Prozessausführungen zu betrachten. Das Prozessverhalten unter bestimmten, bereits eingetretenen Bedingungen, zum Beispiel bei Krankheitsfall oder Ausfällen, lässt sich hiermit genauer bestimmen und verbessern (Zahoransky et al. 2015). Die gefundenen Ergebnisse sind somit Teil der Überwachungsphase und sind bei der Neugestaltung oder Optimierung von Geschäftsprozessen hilfreich.

[4]Erstellt mit Disco (Günther et al. 2012)

2.2.2 Auswirkungen von Prozessänderungen erkennen

Eine große Rolle in der Überwachungsphase des BPM-Lebenszyklus spielt die Möglichkeit, die Auswirkungen geplanter Anpassungen und Änderungen an den Geschäftsprozessen zu erkennen. Zur Beantwortung dieser Frage werden die Kosten, sowie weitere Faktoren untersucht, um den Nutzen und die Auswirkungen der geplanten Anpassungen abzuschätzen. Solch Abschätzungen finden durch Prozesskostenrechnungen (PKR) oder Leistungskennzahlen statt, welche auch durch die Techniken des PMs aus den Prozess-Logs und durch weitere Prozessbetrachtungen zustande kommen. Die Verfahren für das Beziffern von Prozesskosten sind im Überblick:

- Activity Based Costing (ABC) (Cooper et al. 1988), oder entsprechende Erweiterungen wie das Time-driven Activity Based Costing (TD ABC) (Cooper 1988; Özyürek et al. 2014; Kaplan et al. 2004), zum Errechnen von Prozesskosten an Hand von zeitlichen Parametern.

- Total Cost of Ownership (Krämer 2007), um direkte und indirekte Kosten zu ermitteln.

- Life Cycle Costing (LCC) (Zehbold 1996) zur Betrachtung aller entstehenden Kosten eines Produktes.

Die Betrachtung weiterer Aspekte direkter oder indirekter Kosten erfolgt innerhalb des BPM-Lebenszyklus durch Key Performance Indicator (KPI), zu deutsch Leistungskennzahlen (Parmenter 2015). Diese beschreiben verschiedene Eigenschaften eines Prozesses, z.B. Durchlaufzeiten, Rückrufe oder Sicherheitsverletzungen. PM-Ansätze erlauben das Ermitteln dieser Indikatoren an Hand der Prozess-Logs. Die Auswirkungen umgesetzter Prozessänderungen lassen sich an diesen Indikatoren quantifizieren.

Die nach einer Prozessänderung ermittelten Leistungskennzahlen und Kosten lassen sich mit früheren Kennzahlen vergleichen, um die Erfolge einer Maßnahme festzustellen (Holterman 2013). Dies stellt eine *a posteriori*-Kontrolle dar. Erst nach der Ausführungsphase eines Prozesses innerhalb des BPM-Lebenszyklus sind diese Kontrollen durch Leistungskennzahlen für den neu gestalteten Prozess möglich.

Um den Einfluss von Prozessanpassungen bereits in der Entwurfsphase auf die Leistungskennzahlen und Kosten zu quantifizieren, entstanden Simulationsverfahren, um Auswirkungen durch Prozessänderungen zu schätzen (April et al. 2006; Barnett 2003). Hiermit lassen sich bereits zur Entwicklungszeit unterschiedliche Prozessvariationen miteinander vergleichen und deren Leistungskennzahlen mit dem laufenden Prozess abgleichen. Die genutzten Simulationsmodelle erlauben, Auswirkungen von Prozessanpassungen auf die Leistungskennzahlen noch vor der Ausführung des neuen Prozessmodells abzuschätzen (April et al. 2006). Wechselwirkungen mit anderen Prozessen oder die Auswirkungen sicherheitsrelevanter Änderungen, wie zum Beispiel einer Änderung der Zugriffskontrolle, sind jedoch nicht Teil dieser Simulationen. Die genutzten Modelle abstrahieren und vernachlässigen Konzepte der gegenseitigen Ressourcennutzung von Prozessen, der Zugangskontrollen oder Funktionstrennungen sowie temporale Aspekte. Hierdurch lassen sich veränderte Sicherheits- und Compliancemaßnahmen bisher nicht vollständig quantifizieren, wie in Kapitel 3 genauer erläutert.

2.3 Eingliederung der Arbeit in den BPM-Lebenszyklus

BPM betrachtet Sicherheitsmerkmale wie die Autorisierung, Funktionstrennung sowie Isolation von Prozessen, kann auf Grund der Fokussierung auf einzelne Prozesse und zu großer Abstraktion der Simulationsmodelle jedoch die Wechselwirkung mit weiteren Prozessen nicht ausreichend prognostizieren (Aalst 2010). Die Auswirkung, gerade von Sicherheits- und compliancerelevanten Anpassungen auf zukünftige Prozessabläufe lässt sich daher mit bestehenden Verfahren innerhalb der Entwurfsphase des BPM-Lebenszyklus bisher nicht ganzheitlich abschätzen. Erst nachdem Änderungen an Prozessen gemacht wurde zeigen sich die realen Auswirkungen in der Überwachungsphase des BPM-Lebenszyklus. Die vorliegende Arbeit erweitert daher bestehende Simulationsverfahren, um innerhalb der Entwurfsphase die Wechselwirkung von Sicherheit auf die Verfügbarkeit der Prozesse im Sinne der Stakeholder zu zeigen. Wie in der Einleitung gefordert, wird Prozessentwicklern somit die Möglichkeit gegeben, Sicherheits- und Complianceumsetzungen mit möglichst geringem Einfluss auf die Verfügbarkeit der Prozesse umzusetzen, damit verschieden Interessengruppen, wie z.B. die Unternehmensführung, diese Maßnahmen nicht ablehnen und es zu keinen erhöhten Kosten oder Reputationsschäden durch verzögerte Prozessausführungen kommt (Hendricks et al. 2005).

2.4 Anforderungen an sicherheits- und complianceorientierte Simulationen

Zur Unterstützung des Geschäftsprozessmanagements in diesem Bereich der Sicherheit und Compliance durch die Geschäftsprozesssimulation ist die Betrachtung von nebenläufigen, sich gegenseitig beeinflussenden Prozessen notwendig. Zusätzlich ist es erforderlich, den Kontrollfluss des Prozesses, Einschränkungen durch die Zugriffskontrolle sowie temporale Aspekte zu berücksichtigen. Hierdurch lassen sich die Wechselwirkungen mit anderen Prozessen modellieren, da Sicherheitsbedingungen die Abhängigkeiten zwischen Prozessen und Aktivitäten zusätzlich verstärken (Holderer et al. 2015). Zusammengefasst sind die Anforderungen an ein Simulationsmodell in Abbildung 2.7 dargestellt.

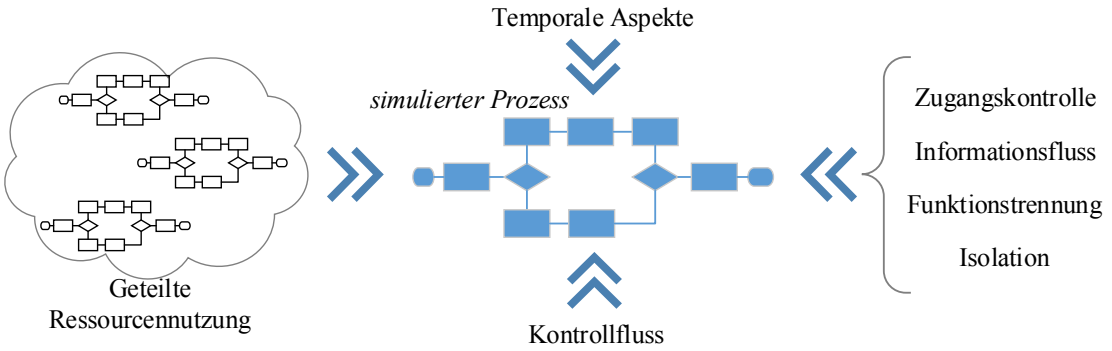


Abbildung 2.7.: Anforderungen an eine sicherheitsorientierte Geschäftsprozesssimulation

Sowohl strukturelle Maßnahmen durch die organisatorischen Ebene, als auch Änderungen der Zugangskontrolle und des Informationsflusses durch die IT-Infrastruktur zeigen Auswirkungen auf das Prozessverhalten, und den Ressourcenbedarf. So wird zum Beispiel durch Sicherheitsbestimmungen, wie einem Vier-Augen-Prinzip oder der Funktionstrennung, die Menge der zur Verfügung stehenden Personen zur Abarbeitung von Aktivitäten reduziert (Holderer et al. 2016). Dies zeigt, dass die Betrachtung der Zugangskontrolle, Funktionstrennung und Isolation ein notwendiger Bestandteil zur Quantifizierung der Auswirkungen von Sicherheitsmaßnahmen auf die Verfügbarkeit von Prozessen ist.

Unterschiedliche Ressourcen werden von den Aktivitäten der Prozesse für gewisse Zeitintervalle genutzt. Diese beschreibt den Grad der Abhängigkeit der Prozesse

untereinander. Für die Dauer der Nutzung stehen Ressourcen für andere Prozesse nicht mehr zur vollen Verfügung. Die Betrachtung dieses Aspekts darf nicht abstrahiert werden, um repräsentative Ergebnisse zu erzielen. Im Rahmen dieser Arbeit wird in Kapitel 4.8 aufgezeigt, wie Verfahren, ähnlich dem PM, einsetzbar sind, um diese Abhängigkeiten automatisiert zu erfassen.

Für eine Modellierung des Problems und der Simulation der Auswirkung von Sicherheits- und Compliancmaßnahmen auf Prozessarchitekturen ist es notwendig, die Struktur der Prozesse selbst, deren Zugriffskontrollen, die Ressourcennutzung sowie temporale Aspekte zu modellieren. Innerhalb des folgenden Kapitels werden bestehende Arten der Geschäftsprozesssimulation beschrieben und miteinander im Hinblick auf die gestellten Anforderungen verglichen.

Kapitel 3

Geschäftsprozesssimulation zur Erkennung der Auswirkung sicher gestalteter Prozesse

Die Simulation ist eine weit verbreitet eingesetzte Methode, um das Verständnis von dynamischen Systemen zu begreifen, den Ausgang von Experimenten abzuschätzen oder Prognosen zu stellen (Aalst 2015). In den späten 60er Jahren begann verstärkt die Ausweitung von Simulationen auf Geschäftsprozesse (Aalst 2015; Schonenberg 2009) und die Umsetzung der gewonnenen Ergebnisse zur Verbesserung von Prozessen (Tumay 1995). Je nach Simulationsart lassen sich gezielt Szenarien testen, um Erkenntnisse zu gewinnen, wie sich Prozesse unter verschiedensten Bedingungen verhalten oder unter welchen Bedingungen der Prozess noch wie gewünscht ausführbar bleibt (Kalibatiene et al. 2016). Weitere Simulationen dienen der Bewertung von Prozessänderungen oder der Ermittlung des Ressourcen- oder Zeitverbrauchs. Innerhalb des BPM-Lebenszyklus setzen die eingesetzten Simulationsverfahren somit ihren Fokus auf das Prozessdesign und die Überwachung.

Es entstand ein neues Forschungs- und Arbeitsfeld innerhalb von BPM: Business Process Re-engineering (BPR). BPR als neue Teildisziplin umfasst die Umarbeitung von Prozessen (Johansson 1993). Diese neu entstandene Disziplin wurde von der Industrie in der Hoffnung, bestehende Prozesse zu verbessern, aufgenommen. Nur 30 % der durchgeführten Verbesserungsmaßnahmen verzeichneten allerdings Erfolge. Dies lässt sich auf mangelnde Datenbasis zurückführen (Hammer et al. 2009). Um die Leistung von geänderten und verbesserten Prozessen zu beurteilen, wurden Tabellen und Flussdiagramme eingesetzt. Dies führte zu

übertrieben optimistischen Schätzungen. Die prognostizierten Kostenersparnisse, verbesserte Durchlaufzeiten oder Servicelevel wurden in der Realität nicht erreicht. Für eine solche manuelle Analyse waren die Geschäftsprozesse zu komplex. Erst der Einsatz von Simulationen ermöglichte die Abschätzung der durch BPR vorgeschlagenen Änderungen. Dadurch ließen sich gezielter Prozesse mit Verbesserungspotenzial erkennen. Zusätzlich vereinfachte dies die Suche nach Optimierungsmöglichkeiten innerhalb des Prozesses und befähigte die Unternehmensführung die einzelnen Optionen miteinander zu vergleichen.

In diesem Kapitel wird die Einsatzfähigkeit bestehender Verfahren zur Simulation von Auswirkungen sicherheitsbezogener Prozessanpassungen geprüft.

3.1 Bausteine der Geschäftsprozesssimulation

Eine Simulation umfasst ein Modell des zu simulierenden Systems, welches mit Informationen über die konkret zu simulierende Instanz erzeugt wird. Hieraus berechnet die Simulation die Dynamik des Systems, um dessen Ausgang zu prognostizieren. Die Prognosen lassen sich auf unterschiedliche Weisen nutzen, zum Beispiel für die Verbesserung des simulierten Systems oder der Steuerung einzelner Teilsysteme.

Simulationen befinden sich je nach Einsatzzweck und System auf differenzierten Abstraktionsebenen. So benötigen Physiksimulationen ein Modell der physikalischen Gesetze, bestehend aus Formeln, welche die Interaktion der Umwelt beschreiben. Sozialwissenschaftler interessieren sich für das Verhalten großer Menschenmengen. Dieses Verhalten lässt sich nicht simulieren, indem die physikalischen Gesetze des menschlichen Organismus angewandt werden. Stattdessen wird hier der einzelne Mensch auf Parameter reduziert, die sein Verhalten darstellen. Geschäftsprozesssimulationen sind wiederum einer anderen Abstraktion unterlegen. Es werden weder die realen Aktionen der Prozesse als solche simuliert, noch wird der Prozess selbst abstrahiert. Stattdessen wird die Abarbeitung der Aktivitäten eingehalten und die Ausführung der einzelnen Aktivitäten an Hand verschiedener Kennzahlen wie Ressourcenverbrauch, entstehende Kosten oder der benötigten Zeitdauer durchgespielt. Auf dieser Abstraktionsebene sind sowohl effiziente Simulationen, als auch aussagekräftige Ergebnisse möglich. Nach (Tumay 1995) besteht

eine Simulation von Geschäftsprozessen mit der geforderten Abstraktionstiefe aus vier Bausteinen:

- Flussobjekte sind die Entitäten eines Prozesses, welche durch Ressourcen und Aktivitäten verarbeitet werden. Sie folgen dem Prozess entlang des Kontrollflusses.

- Ressourcen stellen die Mitarbeiter, Hilfsmittel und Einheiten im Allgemeinen dar, mit deren Hilfe die Aktivitäten zum Wertbeitrag an den Flussobjekten beitragen.

- Aktivitäten sind durch Kontrollflüsse miteinander verbundene Aufgaben, welche durch die Verwendung von Ressourcen die Flussobjekte bearbeiten. Den Aktivitäten können weitere Attribute hinterlegt sein, wie die für die Bearbeitung notwendigen Zeiteinheiten oder den durch die Abarbeitung entstehenden Kosten.

- Kontrollflüsse bestimmen, in welcher Reihenfolge Aktivitäten abgearbeitet werden. Sie verbinden die Aktivitäten miteinander, womit sichergestellt wird, welche Aktivität oder Aktivitäten aufeinander folgen. Kontrollflüsse können Verzweigungen als auch parallele Ausführungspfade enthalten.

Ein zur Simulation von Geschäftsprozessen zu Grunde liegendes Modell muss in der Lage sein, diese Bausteine, aus denen ein Geschäftsprozess besteht, abzubilden. Hierdurch kann die Simulation dem Prozess entsprechend ablaufen und sinnvolle Ergebnisse erzielen.

3.1.1 Simulationsformen

Innerhalb der Simulation von Geschäftsprozessen existieren verschieden ausgeprägte Modelle und Simulationsmethoden zur Behandlung unterschiedlichster Fragestellungen. Die vorhandenen Arten von Geschäftsprozesssimulationen lassen sich in Kategorien einteilen (Tumay 1995):

Kontrollfluss basierte Simulation

Diese Simulationsart verfolgt die Abarbeitung innerhalb eines gegebenen Prozessmodells entsprechend des Kontrollflusses. Gemäß der hierdurch auferlegten Aktivitätsreihenfolge wird die Ausführung jeder Aktivität simuliert. Dabei lassen sich zum Beispiel Ressourcenverbrauch oder Kosten eines Prozesses ermitteln. Diese Simulationsart spiegelt den Prozess exakt wider und orientiert sich strikt an dem Prozessmodell.

System Dynamics

System Dynamics ist eine Methode der dynamischen Simulation von Modellen (Forrester 1997). Diese Simulationsart wird nicht ausschließlich bei Geschäftsprozessen eingesetzt, sondern auch zum Beispiel zur Analyse in den Sozialwissenschaften. System Dynamics bezeichnet die Simulation abstrakter Werte an Hand von mathematischen Modellen (Sterman 2000). Dabei wirkt die Veränderung einer Variable positiv oder negativ auf andere Variablen. Welche Variablen wie mit welchen anderen interagieren, wird durch das zu Grunde liegende Modell beschrieben. Auch wenn die Simulation nicht die direkte Abarbeitung der Aktivitäten gemäß eines Prozessmodells simuliert, kann dieses Verfahren Kennzahlen über den Ressourcenverbrauch, Kosten oder Periodizitäten im Durchlauf von Prozessen erkennen lassen. Ein Beispiel für eine solche Simulationssoftware ist PowerSim (Schöneborn 2013).

Agentenbasierte Simulation

Das Einsatzgebiet von agentenbasierter Modellierung und Simulation sind verknüpfte, miteinander agierende Systeme (Wooldridge 2009). Das Verhalten einer Gesamtmenge vieler Individuen lässt sich durch die agentenbasierte Simulation voraussehen. Diese Multiagantensysteme sind daher geeignet, beispielsweise soziale Netzwerke oder Wirtschaftsysteme zu simulieren (Kita et al. 2016). Gebäude, Städte, Verkehrsnetze und weitere Systeme lassen sich mit diesen Erkenntnissen des Gesamtverhaltens einer großen Menge planen und Abweichungen des normalen Verhaltens werden frühzeitig erkennbar (Schadschneider et al. 2011). Zwar agieren

auch Geschäftsprozesse durch ihre Ressourcen- oder Informationsnutzen miteinander, doch "handeln" Geschäftsprozesse nicht entsprechend der Vorgaben eines Multiagentensystems. Agenten innerhalb der Simulation sind als autonom agierende Einheiten mit eigener Wahrnehmung, Handlungsspielraum, Zielen und Interaktionen beschrieben. Im Gegensatz hierzu folgen Geschäftsprozesse einem fest definierten Kontrollfluss. Die agentenbasierte Simulation entspricht daher nicht der Charakteristik von Geschäftsprozessen.

Diskrete Simulation

Diskrete, auf Ereignissen basierte Simulationen führen die Simulation einzelner Aktivitäten in diskreten Zeitschlitzen durch. Das Verfahren ist analog zu den auf Kontrollfluss basierenden Simulationen. Unterschiede bestehen in der simulierten Abarbeitung der Aktivitäten. Die Abarbeitung erfolgt in diesem Fall gemäß einer durch das Modell bestimmten zeitlichen Komponente. Einzelnen Aktivitäten sind Zeitdauern hinterlegt um die Abarbeitungszeit der Ausführung zu simulieren. Hierdurch entspricht die simulierte Abarbeitung des Prozessmodells möglichst genau der realen Abarbeitung. Die diskrete Simulation erlaubt es, Wartezeiten zu erkennen, zum Beispiel verursacht durch langsame Aktivitäten oder lange Pfade, die erst durchlaufen werden müssen, bevor weitere Aktivitäten beginnen können. In den 80er Jahren vorgestellte und aktiv weiter verfolgte Implementierungen sind SIMPROCESS (Swegles 1997) oder Extend+BPR (Krahl 2001).

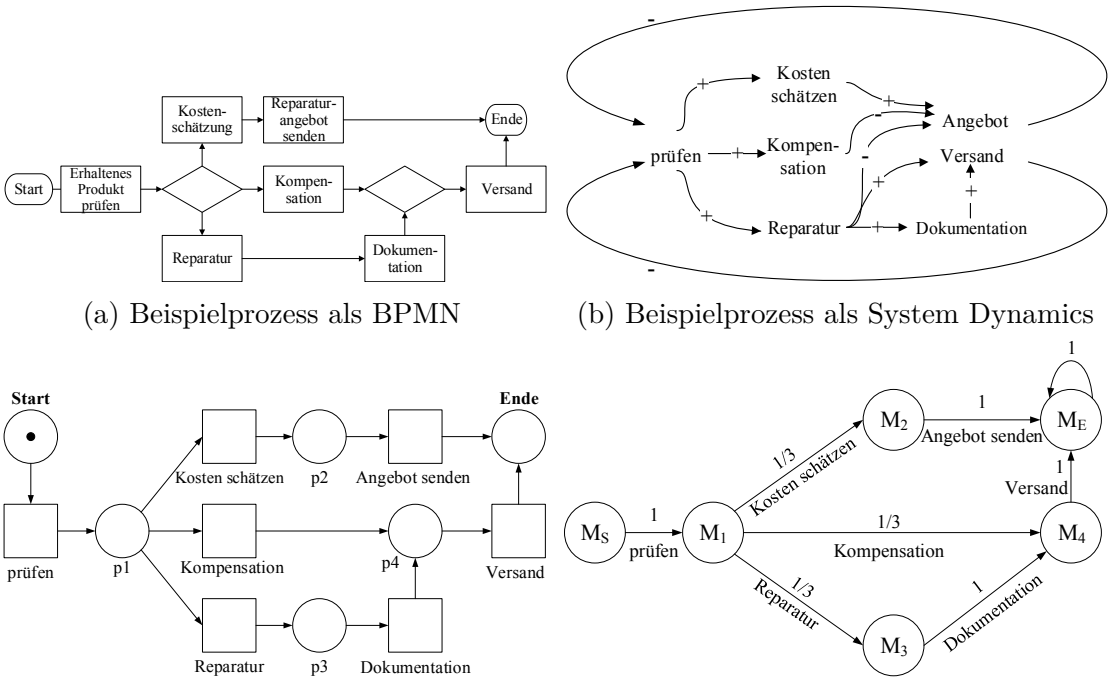
3.1.2 Simulationsmodelle

Entsprechend der Simulationsart müssen die Geschäftsprozesse in unterschiedlichen Modellen vorliegen, um diese der Simulation zugänglich zu machen. Verbreitete Modelle sind Markow-Ketten, Petri-Netze oder BPMN-Modelle. Weitere Modellierungssprachen sind UML-Aktivitätsdiagramme oder proprietäre Lösungen wie die Beschreibungssprache Yet Another Workflow Language (YAWL) (Kluza et al. 2016; Aalst et al. 2005).

Häufig eingesetzte Modelle zur Simulation sicherheitsrelevanter Eigenschaften sind Markow-Ketten (Howard 2012) oder Petri-Netze (Aalst 1996). Der Einsatz von System Dynamics zur Analyse statischer Eigenschaften benötigt eine Darstellung der Geschäftsprozesse als Satz zusammenhängender Variablen (Sterman 2000).

Die zur grafischen Modellierung genutzte BPMN-Sprache ist intuitiv verständlich, zeigt jedoch Schwächen in ihrer Semantik und weist Mehrdeutigkeiten auf (Dijkman et al. 2008). BPMN 2.0 als Nachfolge von BPMN, besitzt eine eindeutige Semantik, welche auf Petri-Netze basiert und mit diesen übereinstimmt (Allweyer 2015; Aalst 2015).

In Abbildung 3.1 sind die vorherrschenden Modelle grafisch dargestellt. Alle Modelle bilden den gleichen, beispielhaften Kompensationsprozess ab: Für ein defektes Produkt wird geprüft, ob ein Garantieanspruch besteht, woraufhin es repariert oder ersetzt wird. Ist es kein Garantiefall, wird ein Kostenvoranschlag für die Reparatur versendet.



(a) Beispielprozess als BPMN

(b) Beispielprozess als System Dynamics

(c) Beispielprozess als Petri-Netz

(d) Beispielprozess als Markov-Kette

Abbildung 3.1.: Beispielprozess modelliert in unterschiedlichen Sprachen

BPMN

Abbildung 3.1a zeigt den Kompensationsprozess als BPMN-Modell. Diese Sprache wird im Rahmen von BPM hauptsächlich eingesetzt um Prozesse abzubilden, neu zu gestalten oder bestehende Prozesse digital zu erfassen. Die Modellierungssprache erlaubt eine grafische Bearbeitung. Die Sprache wurde entwickelt, damit sie intuitiv auch für Benutzer ohne technischen Hintergrund verständlich

ist (White et al. 2004). Das entstandene, grafische Modell bietet jedoch keine mathematische Semantik. Ohne zusätzliche Informationen können dadurch BPMN-Modelle mehrdeutig sein (Dijkman et al. 2008). Dies hinderte die Entwicklung von Methoden zur Analyse von als BPMN modellierten Geschäftsprozessen.

BPMN 2.0-Modelle umgehen die Mehrdeutigkeit. Diese Modelle bieten die Möglichkeit, die Ausführungssemantik durch ein zusätzliches Petri-Netz zu definieren (Allweyer 2015). Die Betrachtung von Petri-Netz-Modellen ist damit auch auf BPMN 2.0-Modelle übetrag- und anwendbar.

System Dynamics

System Dynamics-Modelle, wie grafisch in Abbildung 3.1b gezeigt, modellieren nicht den Ablauf von Aktivitäten als solche, sondern die Wechselwirkung zwischen einzelnen Aktivitäten (Sterman 2000). So können sich Aktivitäten positiv oder negativ auf andere Aktivitäten oder Variablen aus dem Modell auswirken. Die Aktivität "Reparatur" zum Beispiel würde sich negativ auf den Lagerbestand auswirken, jedoch positiv auf die Kundenzufriedenheit. Mit einem System Dynamics-Modell lassen sich Wechselwirkungen, auch von abstrakten Gegenständen außerhalb des Prozesses, sehr gut simulieren. Jedoch ist das konkrete Prozessverhalten wegen den fehlenden direkten Kontrollflüssen durch System Dynamics nicht simulierbar.

Markow-Ketten

Markow-Ketten erlauben die Modellierung stochastischer Prozesse (Howard 2012). Damit eignet sich diese Modellierungssprache auch zur Darstellung von Geschäftsprozessen. In Abbildung 3.1c ist der Beispielprozess als Markow-Kette dargestellt. Jede Aktivität überführt die Markow-Kette in einen anderen Zustand, von dem aus, je nach aktuellem Zustand, weitere Aktivitäten ausführbar sind. In Markow-Ketten-Modellen muss jedem Übergang von einem Zustand in einen anderen Zustand eine Wahrscheinlichkeit zugeordnet sein. Der Endzustand oder Terminalzustand wird modelliert, indem die einzig ausgehende Zustandsänderung mit einer Wahrscheinlichkeit von 1 wieder auf den gleichen Endzustand zeigt (Zustand M_E). Damit verbleibt die Markow-Kette stabil in diesem Zustand.

Markow-Ketten bilden Zustände explizit ab. Innerhalb von Geschäftsprozessen können allerdings parallele Ausführungspfade existieren, welche zu einem sprunghaften Anstieg der möglichen Zustände führen. Die Abbildung auf einzelne Zustände ist dadurch nicht mehr praktikabel, da diese hierdurch schnell anwachsen. In Abbildung 3.2 wird ein Beispielprozess mit einer parallelen Abarbeitung von vier Aktivitäten in eine Markow-Kette überführt. Die Aktivitäten B, C, D und F lassen sich nebenläufig abarbeiten. Dies führt zu einer großen Menge neuer Zustände, da die explizite Abarbeitungsreihenfolge aus einer Permutation der Mengen $\{B, C\}$ und $\{D, F\}$ besteht. Die explizit gezeigten Zustände vergrößern das Markov-Ketten-Modell soweit, dass die Verständlichkeit nicht mehr gewährleistet ist. Sollen weiterhin Ressourcen oder zeitliche Eigenschaften in das Modell aufgenommen werden, steigern sich die vorhandenen Zustände weiter.

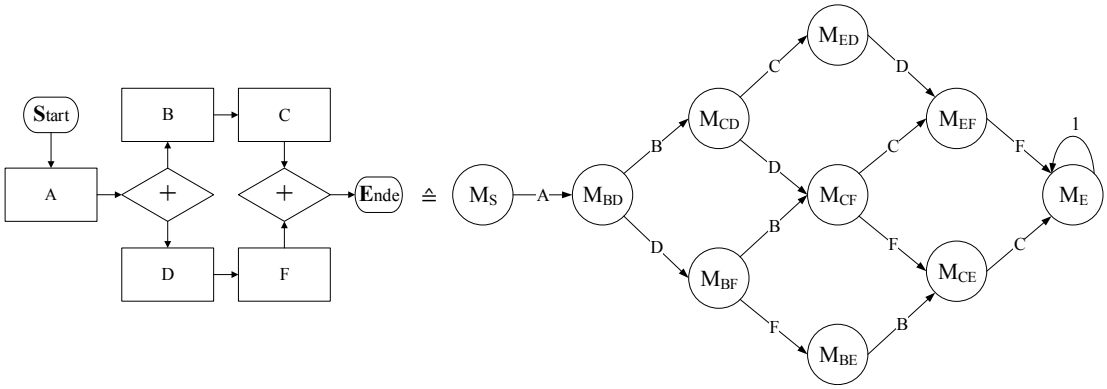


Abbildung 3.2.: Beispielhafte Vergrößerung eines Markow Modells durch parallele Ausführungspfade B-C und D-F.

Markow-Ketten eignen sich mit ihrer Struktur für computergestützte oder mathematische Analysen (Häggström 2001), ermöglichen jedoch keine an BPMN angelehnte, intuitive Modellierung von Geschäftsprozessen. Die Anwendbarkeit ist zwar gegeben, allerdings nicht praktikabel, da jeder Zustand explizit zu modellieren ist.

Petri-Netze

Petri-Netze (PN) sind eine Modellierungssprache, welche sowohl Zustände, als auch Kontrollfüsse darstellt. Die Zustände werden nicht explizit durch Knoten angegeben, sondern sind innerhalb des Modells, in sogenannten Plätzen, kodiert. Das vorherige Beispiel aus Abbildung 3.2 in ein Petri-Netz überführt zeigt nicht den

sprunghaften Anstieg der Modellgröße, wie zuvor für Markow-Ketten. Abbildung 3.3 zeigt den gleichen Prozess als Petri-Netz-Modell. Die Beschreibung der Elemente eines Petri-Netzes folgen im nächsten Abschnitt.

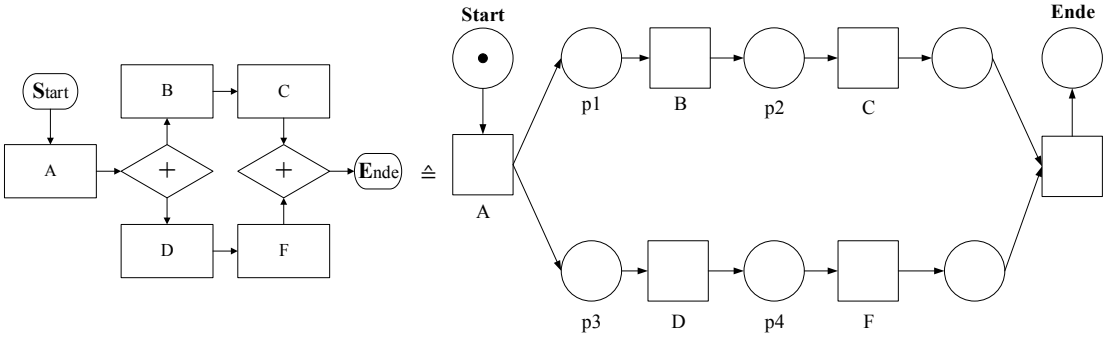


Abbildung 3.3.: Parallele Ausführungspfade modelliert als Petri-Netz

Petri-Netze erlauben die Darstellung sowohl paralleler, als auch sich gegenseitig ausschließenden Abarbeitungspfade. Zusätzliche Aktivitäten oder Verzweigungen innerhalb der parallelen Pfade führen somit nicht zu einem überproportionalen Anstieg der Modellgröße. Auch mit der Verwendung weiterer, häufig in BPMN eingesetzten Konstrukte bleibt das entsprechend überführte Petri-Netz-Modell verständlich, da die Konstrukte sich direkt in Petri-Netze abbilden lassen (Muehlen et al. 2008). Dies führte dazu, dass Petri-Netze auch in der verbreiteten Prozessbeschreibungssprache BPMN 2.0 ihren Einsatz zur Definition des Kontrollflusses fanden. Hiermit zeigt sich die Verwendbarkeit von Petri-Netzen für die Modellierung von Geschäftsprozessen. Sie werden in dieser Arbeit als Grundlage der Simulation von Geschäftsprozessen herangezogen.

Weitere Modellierungssprachen

Die Modellierung von Geschäftsprozessen erfolgt auch durch weitere Modellierungssprachen. Diese Sprachen sind ereignisbasiert, wie zum Beispiel ereignissgesteuerte Prozessketten (Nüttgens et al. 2002) oder die Integrated Definition-Methode (Mayer et al. 1992). Andere Sprachen wurden für konkrete Einsatzzwecke spezialisiert entwickelt, wie zum Beispiel UML-Aktivitätsdiagramme (Dumas et al. 2001) oder die Prozessbeschreibungssprache YAWL (Aalst et al. 2005). Im ersten Fall, der ereignisbasierten Sprachen, ist die

Abfolge von Aktivitäten nicht konkret gegeben. Solche Sprachen eignen sich konzeptionell nicht für die Erweiterungen um eine Zeit- und Ressourcendimension (Kluza et al. 2016), weswegen sie sich nicht zur Simulation von Prozessen eignen. Die Klasse der spezialisierten Modellierungssprachen verlieren durch ihre Ausprägungen an Allgemeingültigkeit, da sie sehr spezifisch ihrem Verwendungszweck angepasst wurden.

3.2 Modellierung von Prozessen mit Petri-Netzen

Petri-Netze, entwickelt in den 60ern Jahren von Carl Adam Petri (Petri 1962), sind in unterschiedlichen Ausprägungen immer noch Grundlage aktueller Forschungen (Reisig 2012). Die Eignung von Petri-Netzen zur Modellierung von Geschäftsprozessen ist durch ihre Struktur gegeben, da Petri-Netze sowohl Zustände als auch Kontrollflüsse abbilden.

In (Zahoransky et al. 2016) sowie (Accorsi et al. 2011a) wird demonstriert, dass sich Geschäftsprozesse als Petri-Netze darstellen lassen, da alle vier wesentlichen Elemente eines Geschäftsprozesses als Petri-Netz abbildbar sind. Sie ermöglichen die Modellierung von Flussobjekten, Ressourcen, Aktivitäten und den zugehörigen Kontrollflüssen[1]. Es ist mit Petri-Netzen möglich, die Aneinanderreihung von Aktivitäten, Verzweigung von Aktivitäten, die parallele Abarbeitung oder der Zusammenschluss von Arbeitsschritten zu modellieren. Die entsprechenden Petri-Netz Artefakte sind mit den dazu entsprechenden BPMN Elementen in Abbildung 3.4 zu erkennen.

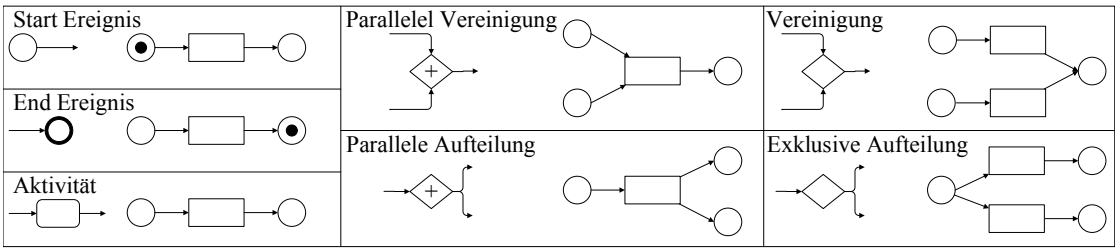


Abbildung 3.4.: BPMN Prozess-Artefakte als Petri-Netz-Elemente[2].

Mit Petri-Netzen lassen sich Kontroll- und Informationsflusseigenschaften prüfen, da im Gegensatz zu der Darstellung in BPMN keine Mehrdeutigkeiten existieren.

[1] Vergleiche Absatz 3.1
[2] Grafik nach Von Stackelberg et al. 2014

Petri-Netze eigenen sich also auch zur Analyse sicherheitskritischer Eigenschaften. In Arbeiten wie (Von Stackelberg et al. 2014) zeigt sich, dass sich als BPMN eindeutig definierte Geschäftsprozesse algorithmisch in Petri-Netz-Modelle transferieren lassen. Die resultierenden Modelle werden in weiteren Arbeiten zur Analyse von Daten- und Informationsfluss genutzt (Accorsi et al. 2014) und um Sicherheitsanforderungen zu prüfen (Accorsi 2013). Weiterhin finden Petri-Netze den Einsatz bei der Effizienzsteigerung oder der Berechnung des Ressourcenverbrauchs von Geschäftsprozessen (Zahoransky et al. 2016; Li et al. 2004; Teixeira et al. 2015). Die allgemein gültige Eignung von Petri-Netzen zur Prozessmodellierung und Prozessverifikation wird durch Arbeiten wie (Accorsi et al. 2011b) gezeigt.

Aalst[3] zeigt in mehreren Arbeiten ebenfalls die Eignung von Petri-Netzen zur Darstellung von Geschäftsprozessen. Der Autor berichtet über seine Erfahrungen bei der Arbeit des Projektes *Sagitta-2000* (Aalst 1998b). Das Projekt zur Entwicklung eines verteilten Informations- und Workflow-Systems zur Steuerung und Unterstützung der nationalen Zollabfertigung wurde 1994 von der dänischen Zollbehörde in Auftrag gegeben. Innerhalb des Projektes wurden bestehende Workflow-Systeme an Hand der Ergebnisse einer Anforderungsanalyse evaluiert. Die durchgeführte Anforderungsanalyse kam zu dem Ergebnis, dass Petri-Netz basierte Systeme die notwendigen Funktionen bieten. Der Autor überträgt seine Anforderungen auf Geschäftsprozessmodelle und schließt daraus auf die allgemeine Eignung von Petri-Netzen zur Darstellung von Geschäftsprozessen.

Diese und weitere Arbeiten führen Prozesssimulationen mit erweiterten Petri-Netzen (Murata 1989) aus. Diese Erweiterungen sind beispielsweise Colored Petri Nets (CPN) zur Betrachtung von Ressourcen, Timed Petri Nets (TPN) zur Simulation zeitlicher Eigenschaften (Merlin et al. 1976) oder Information-Flow Petri Nets (IF-Net) (Stocker et al. 2013), welche sicherheitsrelevante Betrachtungen ermöglichen.

Um die Eignung der Modelle zur Simulation von Geschäftsprozessen auszuloten, wird in diesem Absatz sowohl die Semantik von Petri-Netzen als auch deren unterschiedliche Ausprägungen definiert.

[3] Vgl. Aalst 1998b.

3.2.1 Einführung Petri-Netze

Petri-Netze bestehen aus endlichen Mengen von Stellen P, Transitionen T, gerichteten Kanten F, optionalen Kantengewichten W und einer Markierung m. In der grafischen Darstellungsweise von Petri-Netzen werden Plätze als Kreise, Transitionen als Rechtecke und Marken als Punkte dargestellt. Die einzelnen Elemente und ihre Synonyme sind in Abbildung 3.5 aufgezeigt. Die Abbildung zeigt ein gültiges, minimales Petri-Netz mit einer Marke, zwei Plätzen, einer Transition und zwei Kanten mit jeweils einem Kantengewicht von 1. Üblicherweise werden Kantengewichte nur angegeben, wenn sie sich von 1 unterscheiden.

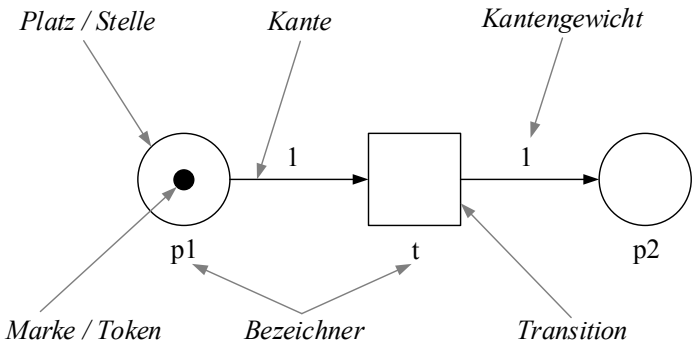


Abbildung 3.5.: Erläuterung der Elemente von Petri-Netzen

In Petri-Netz Modellen verbinden Kanten Plätze mit Transitionen und umgekehrt. Plätze und Transitionen sind stets abwechselnd verbunden. Eine Verbindung von Transition zu Transition oder Platz zu Platz ist unzulässig. Plätze können Marken beinhalten, die durch Transitionen weitergereicht werden. Die Stelle oder Stellen, in denen sich die Marken befinden, kodieren den Zustand m des Netzes, auch Markierung bezeichnet.

Zur Darstellung von Geschäftsprozessen besitzen Petri-Netze einen Quell- und einen Senken-Platz. Dies sind Plätze mit jeweils nur ausgehenden Kanten für die Quelle und eingehenden Kanten für die Senke. Diese Stellen entsprechen dem Start- und dem Endpunkt des Geschäftsprozesses (Aalst 1997). Ein Petri-Netz, welches u.a. dieser Anforderung genügt wird Workflow net (WF-net) genannt.

Formal ist ein Petri-Netz definiert als:

Definition 3.1. *Ein Petri-Netz ist ein Tupel (P,T,F,W,m_0) mit*

- *P als endliche Menge von Stellen*
- *T als endliche Menge von Transitionen*
- *F als Flussrelation ist eine endliche Menge gerichteter Kanten* $F \subseteq (P \times T) \cup (T \times P)$
- *W als die Kantengewichtung* $W \to \mathbb{N}$ *der Relationen*
- $m_0 \to \mathbb{N}_0$ *als die Startmarkierung.*

Dabei ist die Menge der Transitionen und Plätze disjunkt $P \cap T = \emptyset$ *und nicht leer* $P \cup T \neq \emptyset$*. Die Flussrelation* $F \subseteq (P \times T) \cup (T \times P)$ *verbindet Stellen mit Transitionen und Transitionen mit Stellen mit den Kantengewichten W.* ⊣

Für Petri-Netze weist die Startmarkierung m_0 den Plätzen die anfängliche Anzahl von Marken zu.

Dynamik eines Petri-Netzes

Durch das sogenannte Schalten oder Feuern von Transitionen ändert sich die Markierung des Netzes. Die Anzahl, als auch die Position der vorhanden Marken kann sich hierdurch ändern. Dies stellt die Dynamik in Petri-Netzen dar und überführt das Netz von einem Zustand in einen anderen.

Um die Dynamik in Petri-Netzen besser zu beschreiben, werden zuerst Vorbereich und Nachbereich für Plätze und Transitionen folgendermaßen definiert:

- Der Vorbereich $\bullet x$ mit $x \in P \cup T$ bezeichnet die Elemente (Plätze oder Transitionen), die durch eine Kante zu x verbunden sind: $\bullet x = y | (y, x) \in F$
- Der Nachbereich $x\bullet$ mit $x \in P \cup T$ bezeichnet die Elemente, die durch eine Kante von x aus erreichbar sind: $x\bullet = y | (x, y) \in F$

Die Transitionen verändern den Zustand des Petri-Netzes gemäß der Flussrelation F. Damit eine Transition schalten kann, müssen die folgenden Schaltregeln gelten:

- Eine Transition $t \in T$ heißt aktiv, wenn im aktuellen Zustand m jede der Eingangstellen aus $\bullet t$ ausreichend viele Marken gemäß der Kantengewichtungen der eingehenden Kanten besitzt: $\bullet t \subseteq W$.

- Eine aktive Transition $t \in T$ kann feuern (bzw. schalten). Dabei entfernt die Transition t die durch das Kantengewicht der eingehenden Flussrelation angegebene Menge an Marken aus $\bullet t$ und produziert in den Ausgangsstellen die durch die ausgehende Flussrelation spezifizierte Menge an Marken in $t\bullet$.

- Durch das Feuern einer Transition t ändert sich somit die Markierung m des Netzes in $m' = (m \setminus \bullet t) \cup t\bullet$. Das Feuern einer Transition wird dargestellt als $m \xrightarrow{t} m'$. Die insgesamt vorhandene Anzahl an Marken innerhalb des PNs kann sich durch das Feuern aktiver Transitionen verändern.

Die Reihenfolge in welcher Transitionen nacheinander feuern, kann die erreichbaren Markierungen eines Netzes einschränken, da somit Marken konsumiert werden, die dann nicht mehr bereit stehen, um andere Transitionen feuerbereit zu schalten.

Feuersequenzen in Petri-Netzen

Eine Abfolge von geschaltenen Transitionen wird Feuersequenz genannt.

- Eine Feuersequenz $\sigma \in T^*$ wird als $m \xrightarrow{[\sigma]} m'$ dargestellt.

- Eine Markierung m' heißt erreichbar, wenn eine Feuersequenz $*$ existiert, für die gilt: $m \xrightarrow{*} m'$.

Ein Zustandsgraph, auch Erreichbarkeitsgraph oder Markierungsgraph (MG), ist ein Graph aus der Menge aller im Netz erreichbaren Zustände m^* (Peterson 1977). Eine Kante entspricht der Transition t, die gefeuert werden muss, um den entsprechenden Zustand m' zu erreichen: $m \xrightarrow{t} m'$. Somit entsprechen einzelne Pfade a im Zustandsgraph der Feuersequenz $m \xrightarrow{[a]} m'$ des Netzes. Ein Zustandsgraph kann endlich sein, obwohl das entsprechende Netz Schleifen enthält, die zu unbeschränkt langen Feuersequenzen führen können. Ein Zustandsgraph erlaubt somit eine effiziente Analyse des Petri-Netzes (Mayr 1984; Peterson 1977).

Beispiel

Abbildung 3.6 zeigt ein Beispiel für ein Petri-Netz mit den sechs Plätzen $P = \{Start, p1, p2, p3, p4, End\}$ und vier Transitionen $T = \{t1, t2, t3, t4\}$.

Die Plätze sind über folgende Flussrelation verbunden:

$$F = \{Start \to t1, t1 \to p1, t1 \to p2, p1 \to t2, p2 \to t3, t2 \to p3, t3 \to p4, p3 \to t4, p4 \to t4, t4 \to End\}$$

und besitzen jeweils eine Kantengewicht von 1:

$$W = \{(Start \to t1) \to 1, (t1 \to p1) \to 1, (t1 \to p2) \to 1, ..., (t4 \to End) \to 1.$$

Die Startmarkierung lautet $m_0 = \{Start \to 1\}$ oder $m_0 = \{100000\}$ als Kurzschreibweise für die Plätze $\{Start, p1, ..., p4, End\}$.

Der Platz "*Start*" ist in diesem Netz der Quell-Platz, Platz "*End*" der Senken-Platz.

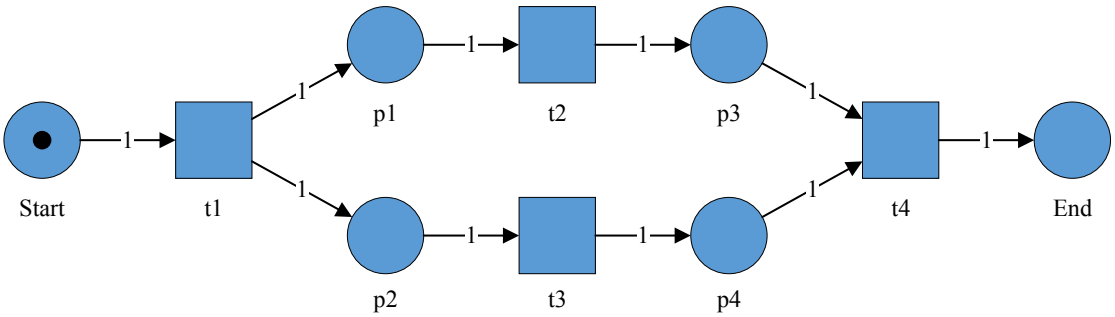


Abbildung 3.6.: Beispiel Petri-Netz.

In diesem Beispiel ist die Transition $t1$ feuerbereit, da alle ihre eingehenden Kanten $\{Start \to t1\}$ gemäß der Kantengewichte $\{(Start \to t1) \to 1\}$ genügend Tokens besitzen: $\bullet t1 = Start$, $m_0 = \{Start \to 1\}$.

Simulation als Token Game

Ein Token Game (Genrich et al. 1981) ist der Fluss der Marken durch das Petri-Netz entsprechend den Feuerregeln. Es ist das wiederholte Schalten von Transitionen, womit das Netz simuliert wird. Wie im gezeigten Netz können mehrere Feu-

ersequenzen existieren, die durch das Token Game "spielbar" sind. Mehrere Pfade führen somit durch das Netz. Das Token Game kann helfen zu erkennen, wie von einem Zustand in einen anderen gewechselt werden kann, oder wie es zu einem unerwünschten Zustand kam. Für ein Petri-Netz, welches einen Geschäftsprozess beschreibt, ist das Token Game die Simulation des Geschäftsprozesses. Wird das Token Game hinreichend oft wiederholt, werden gemäß dem Gesetz der Großen Zahlen alle möglichen Feuersequenzen durchgespielt (Henze 2013). So ist jede mögliche Abarbeitungsreihenfolge des Prozesses abgedeckt. Da das Token Game das Netz entsprechend den Feuerregeln durchläuft, sind so auch Fehler im Kontrollfluss erkennbar.

Der erste Schritt eines Token Games des Beispielnetzes wäre das Schalten der Transition $t1$, da nur diese gemäß den Feuerregeln aktiv ist. Feuert die Transition, produziert sie entsprechend der Relationen $\{t1 \rightarrow p1, t1 \rightarrow p2\}$ und den Kantengewichten $\{(t1 \rightarrow p1) \rightarrow 1, (t1 \rightarrow p2) \rightarrow 1\}$ jeweils einen Token in ihrem Post-Set $t1\bullet = \{p1, p2\}$. Nach dem Feuern der Transition ändert sich somit die Markierung des Netzes zu $m_1 = \{p1 \rightarrow 1, p2 \rightarrow 1\}$. Dieser Zustand ist in Abbildung 3.7 dargestellt.

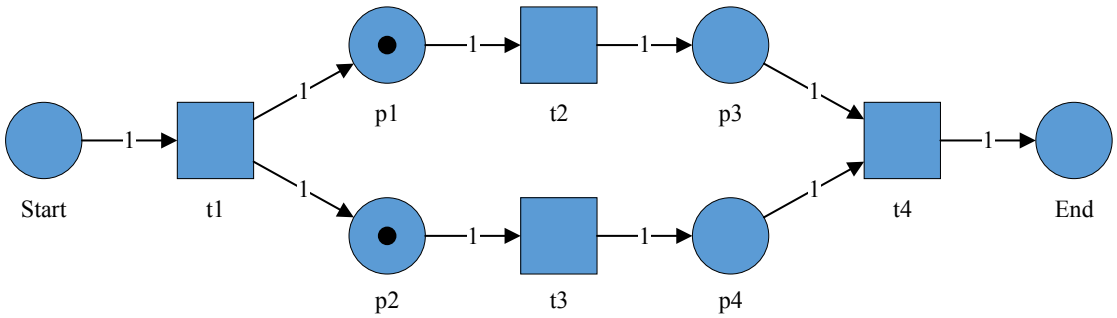


Abbildung 3.7.: Das Beispiel Petri-Netz aus Abb. 3.6 nach dem Feuern der Transition t1.

Aus diesem Zustand können beide Transitionen $t2$ und $t3$ feuern, da beide Transitionen genügend Tokens in ihrem Pre-Set besitzen. Die Transition $t4$ ist jedoch erst aktiv, nachdem sowohl $t2$ als auch $t3$ gefeuert haben und somit Tokens in beide Eingangsplätze von $t4$ produziert haben. Die möglichen Feuersequenzen innerhalb dieses Netzes sind also $t1, t2, t3, t4$ sowie $t1, t3, t2, t4$. Diese lassen sich in einem MG zusammenfassen.

Resultierender Markierungsgraph

Die möglichen Feuersequenzen σ und Zustände m^* sind als Zustandsgraph oder Markierungsgraph in Abbildung 3.8 zusammengefasst. In textueller Form lautet der MG:

$$MG = \{((Start) \xrightarrow{t1} (p1, p2)), ((p1, p2) \xrightarrow{t2} (p2, p3)), ((p1, p2) \xrightarrow{t3} (p1, p4)), ((p2, p3) \xrightarrow{t3} (p3, p4)), ((p1, p4) \xrightarrow{t2} (p3, p4)), ((p3, p4) \xrightarrow{t4} (End))\}.$$

Es zeigt sich für dieses Netz, dass unabhängig der gewählten Feuersequenz das Netz stets im Zustand {000001} mit einer Marke im Platz "End" endet.

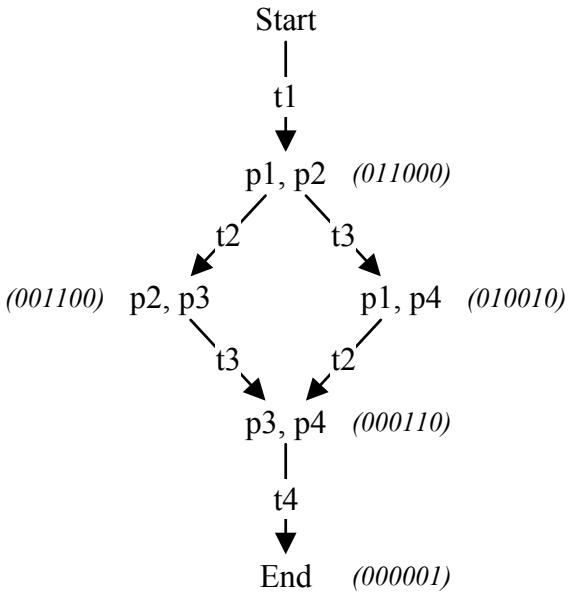


Abbildung 3.8.: Der Zustandsgraph des Petri-Netzes aus Abbildung 3.6.

Die Zustände eines Netzes lassen sich in kurzer Schreibweise darstellen, zum Beispiel kodiert $M = (100000)$ den Zustand, in dem ein Token im Platz "Start" ist und die anderen Stellen leer sind. Jede Position in dieser Schreibweise steht dabei für die Menge an Tokens einer Stelle, in diesem Fall für die Stellen (Start - p1 - p2 - p3 - p4 - End). Diese Schreibweise ist dem MG in Klammern jedem Zustand beigefügt.

MGs erlauben eine effiziente Analyse von Geschäftsprozessen über die Struktur der Zustände, ähnlich den Markow-Ketten. Dies ist insbesondere für die analytische

Auswertung der Netze von Bedeutung, da viele Analysen an einem Zustandsgraphen in linearer Zeit durchführbar sind. Die Erzeugung eines MGs ist jedoch nicht in linearer Zeit generierbar (Chiola et al. 1991).

3.2.2 Colored Petri Nets

Mit Petri-Netzen lassen sich Kontrollflüsse und Aktivitäten darstellen, jedoch keine Ressourcennutzung. Wie anfangs gezeigt, ist jedoch gerade die Ressourcendimension eines Prozesses maßgeblich an der Verfügbarkeit eines Geschäftsprozesses beteiligt (Wildemann 2001; Aalst 1998a). Colored Petri Net (CPN)[4] sind um unterscheidbare Marken erweiterte Petri-Netze (Jensen 1991). Zur Unterscheidung werden den Marken Farben zugeordnet. Neben dem normalen, dem "schwarzen" Token kann ein CPN somit weitere Tokens beinhalten. CPNs stellen durch diese Erweiterung eine Modellierungslösung für die Ressourcennutzung in Geschäftsprozessen dar, da den einzelnen Farben Ressourcen zuweisbar sind. So lässt sich durch CPN modellieren, welche Aktivitäten welche Ressourcen konsumieren und produzieren, sowie die Menge der durch den Prozess genutzten Ressourcen abbilden. Die Simulation eines CPNs als Token Game, also das Durchspielen aller Feuersequenzen gemäß den Feuerregeln gibt zum Beispiel Aufschluss über den maximalen Ressourcenverbrauch oder Ressourcenengpässe innerhalb des simulierten Prozesses.

CPNs erweitern das bestehende Petri-Netz Modell um unterscheidbare Marken, eine Kapazitätsfunktionen, sowie jeweils um Ein- und Ausgabefunktionen.

Die Definition von CPNs lautet:

[4]Im weiteren werden für dieses und weitere Modelle die englischen Eigenname verwendet.

Definition 3.2. *Ein CPN ist ein Tupel N mit N=(P,T,F,$\mathcal{C}$,C,I,O), mit P als endlicher Menge von Stellen, T einer endlichen Menge von Transitionen und F einer endlichen Menge gerichteter Kanten $F \subseteq (F \times T) \cup (T \times P)$. $\mathcal{C}$ ist die Menge der verwendeten Farben. Die Funktion xFy gilt, wenn eine Kante von x nach y existiert. C, I und O sind wie folgt definiert:*

- *Kapazitätsfunktion C: $P \rightarrow N$ ist die maximale Anzahl von Marken einer Stelle pro Farbe*
- *Eingabefunktion I: $T \times P \times C \rightarrow N$ definiert für jede Transition $t \in T$ und jede Stelle $i \in P$ mit iFt und Farbe $c \in \mathcal{C}$ die Anzahl der konsumierenden Marken und Farben.*
- *Ausgabefunktion $O : T \times P \times C \rightarrow N$ definiert für jede Transition $t \in T$ und jede Stelle $i \in P$ mit iFt und Farbe $c \in \mathcal{C}$ die Anzahl der produzierenden Marken und Farben.* ⊣

Der Zustand (die Markierung) eines CPNs $M : P \times C \rightarrow N$ ist die Verteilung der Marken auf die Stellen des Netzes. Ein CPN ist markiert, wenn es als Paar (N,M) angegeben wird.

Die Transitionen im CPN verändern den Zustand des CPNs entsprechend der Flussrelation F durch die folgenden Schaltregeln:

- Eine Transition $t \in T$ heißt aktiv, wenn jede der Eingangstellen ausreichend viele Marken gemäß der Eingabefunktion hat und die Ausgabeplätze genügend Kapazitäten gemäß der Ausgabefunktion und Kapazitätsfunktion haben.
- Eine aktive Transition $t \in T$ kann feuern. Dabei entfernt die Transition t die in der Eingabefunktion hinterlegte Menge an Marken und produziert in den Ausgangsstellen die durch die Ausgabefunktion spezifizierte Menge an Marken.
- Durch das Feuern einer Transition ändert sich die Markierung m des Netzes.

Für CPNs lässt sich ebenso wie für Petri-Netze ein Zustandsgraph erzeugen. So lässt sich zum Beispiel nach der Erzeugung des MGs der maximale Ressourcenbedarf im Endzustand betrachten, oder ob mit den zur Verfügung stehenden Ressourcen der Endzustand erreichbar ist (Jensen et al. 2006; Ratzer et al. 2003).

Beispiel

Abbildung 3.9 zeigt ein Beispiel CPN mit den gleichen Transitionen, Plätzen und Flussrelation wie das Petri-Netz aus Abbildung 3.6. Die Kapazitätsfunktion ergibt sich zu

$$C = \{Start \to (1,1), p1 \to (1,0), p2 \to (1,2), p3 \to (1,0), p4 \to (1,1), End \to (1,0)\}$$

für jeweils die Markenfarbe (Schwarz, Rot). In den Abbildungen sind die Kapazitäten jeweils über den Plätzen aufgetragen.

Die Eingabefunktion ergibt sich zu

$$I = \{(t1, Start, Schwarz) \to 1, (t1, Start, Rot) \to 1, (t2, p1, Schwarz) \to 1, (t3, p2, Schwarz) \to 1, (t3, p2, Rot) \to 1, (t4, p3, Schwarz) \to 1, (t4, p4, Schwarz) \to 1, (t4, p4, Rot) \to 1\}.$$

Die Ausgabefunktion lautet

$$O = \{(t1, p1, Schwarz) \to 1, (t1, p2, Schwarz) \to 1, (t1, p2, Rot) \to 2, (t2, p3, Schwarz) \to 1, (t3, p4, Schwarz) \to 1, (t3, p4, Rot) \to 1, (t3, End, Schwarz) \to 1\}$$

Die Startmarkierung des Netzes ist $m_0 = \{Start \to (1,1)\}$

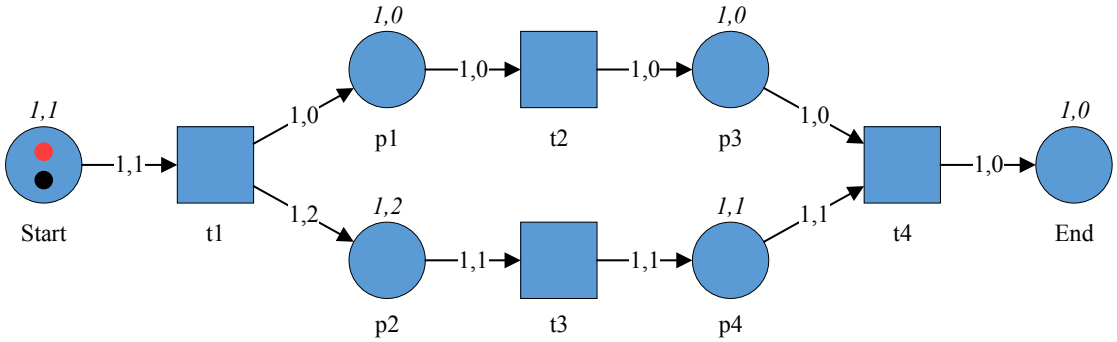


Abbildung 3.9.: Beispiel Colored Petri Net.

Transition $t1$ ist feuerbereit, da genügend schwarze und rote Marken gemäß der Eingabefunktion I im Platz *Start* zu Verfügung stehen und in den darauf folgenden Plätzen genügend Kapazität gemäß der Ausgabefunktion O zur Verfügung steht. Nachdem Transition $t1$ gefeuert hat, ist entsprechend O ein schwarzer Token in

Platz $p1$ und zwei rote sowie ein schwarzer Token in Platz $p2$. Die Markierung ergibt sich dann zu: $m_0 \xrightarrow{t1} m_1$ mit $m_1 = \{p1 \rightarrow (1,0), p2 \rightarrow (1,2)\}$. Die neu entstandene Markierung ist in Abbildung 3.10 visualisiert.

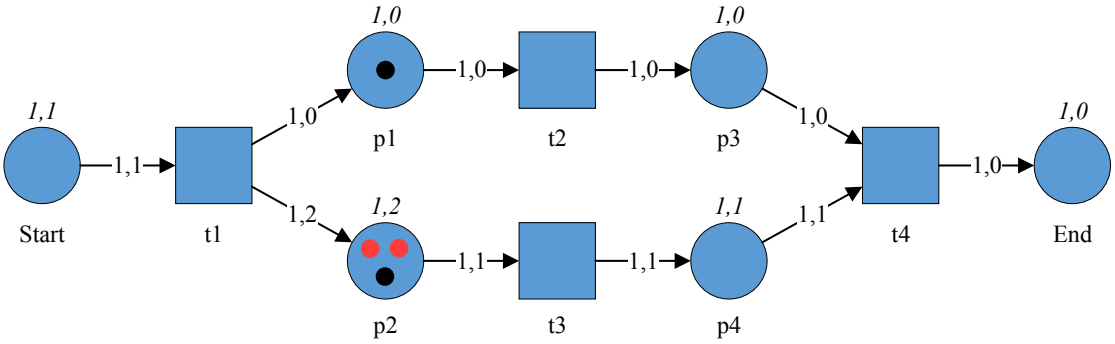


Abbildung 3.10.: Beispiel Colored Petri Net.

Die rote Marke im Beispiel könnte abstrakt eine Datei, Information oder Ressource darstellen, die dupliziert und bearbeitet wird.

3.2.3 Information-Flow Petri Nets

CPNs ermöglichen die Modellierung von Ressourcen und deren Verlauf innerhalb von Prozessen. In realen Geschäftsprozessen ist es jedoch nicht zulässig, dass alle beteiligten Personen auf alle Ressourcen zugreifen oder alle Aktivitäten ausführen dürfen (Wolter et al. 2008). Diese Sicherheitsaspekte von Geschäftsprozessen lassen sich mit bisher gezeigten Modellen nicht nachbilden und somit auch nicht analytisch betrachten.

Aus dieser Lücke heraus entstanden Information-Flow Petri Net (IFNet). Ein IFNet ist eine Erweiterung von CPNs um einen Rechte- und Zugangskontext (Stocker et al. 2013). Zusätzlich weisen IFNets jeder Transition ein Vertraulichkeitslevel zu. Einzelnen Ressourcen und Aktivitäten lassen sich Benutzer zuordnen, welche die Ressource nutzen und die Aktivitäten ausführen dürfen. Damit eignen sich IFNets um sicherheitsrelevante Aspekte von Geschäftsprozessen, wie die Sicherstellung der Vertraulichkeit und des Informationsflusses zwischen den Vertraulichkeitsleveln zu analysieren. Mit IFNets kann somit zum Beispiel sichergestellt werden, dass kein ungewollter Informationsfluss zwischen Mitarbeitern entsteht.

Ein IFNet ist definiert als:

Definition 3.3. *Ein IFnet ist ein Tupel ((P,T,F,$\mathcal{C}$,C,I,O),A,AC,G) mit:*

- *(P,T,F,$\mathcal{C}$,C,I,O) dem zu Grunde liegenden CPN.*
- *A die Zugangsfunktion , welche jeder Transition zuordnet, wie sie auf Objekte zugreift.*
- *AC der Analysekontext, welcher den Transitionen Sicherheitsklassifikationen zuweist.*
- *G ist eine Funktion, die den Transitionen Prädikate zuweist, welche zu wahr oder falsch evaluieren und beeinflusst, ob die Transition schalten darf.* ⊣

Der hinterlegte Analysekontext AC ist folgendermaßen definiert:

Definition 3.4. *AC=(L,$\mathcal{S}_\mathcal{U}$,U) mit*

- *L: $T \to \{high, low\}$ die Labelinformation, zu welcher Kategorie die Transition zugehört.*
- *$\mathcal{S}_\mathcal{U} : T \to U$ ordnet den Transitionen Benutzer zu, welche die Transition ausführen dürfen.*
- *$\mathcal{S}_L : U \to \{high, low\}$ ordnet den Benutzern die Sicherheitslabel $\{high, low\}$ zu. Benutzer mit Sicherheitslabel low dürfen keinen Transitionen mit dem Sicherheitslabel high zugewiesen werden.*
- *U ist die Menge aller möglichen Benutzer* ⊣

Die Zugangsfunktion A ist definiert als:

Definition 3.5. *$A : T_R \times \mathcal{C} \to \mathcal{P}(\mathcal{M}_\mathcal{A})$ weist jeder Transition T_R und jeder Tokenfarbe $\mathcal{C}$ zu, wie die Transition auf dieses Token zugreift: $\mathcal{M}_A = \{read, write, delete, create\}$* ⊣

Die Funktion G gibt an, unter welchen Bedingungen eine Transition schalten darf:

Definition 3.6. *G weist den Transitionen evaluierbare Prädikate zu: $G \in T \to P_\mathcal{C}$. $P_\mathcal{C}$ ist die Menge der Prädikate der eingefärbten Marken. Ein Prädikat evaluiert zu wahr oder falsch.* ⊣

Für das Beispiel $p(rot)$ ist p der Prädikatsname, die Angabe in Klammern gibt die Tokens an, die zur Evaluation von p notwendig sind.

Durch die Funktion G wird die von den CPNs geerbte Feuerregel verändert. Feuerbar ist eine Transition nur, wenn die Feuerregeln gemäß der CPN-Definition erfüllt sind, als auch die Funktion G zu wahr evaluiert. Ist diese Funktion für eine Transition nicht definiert, gelten die CPN-Feuerregeln.

IFNets finden Anwendung durch Softwarepakete wie das Security Workflow Analysis Toolkit (SWAT) (Zahoransky et al. 2016) oder InDico (Accorsi et al. 2011b). Mit diesen Werkzeugen lassen sich musterbasiert Sicherheitsaspekte von Geschäftsprozessen prüfen.

Beispiel

Die Abbildung 3.11 zeigt ein IFNet mit der gleichen Konfiguration wie das CPN aus Abbildung 3.9. Zusätzlich werden den Transitionen Sicherheitslevel zugewiesen:

$$L = \{(t1 \rightarrow high), (t2 \rightarrow low), (t3 \rightarrow hight), (t4 \rightarrow high)\}.$$

Diese Zuweisung ist in der Abbildung den jeweiligen Transitionsnamen in eckiger Klammer beigefügt.

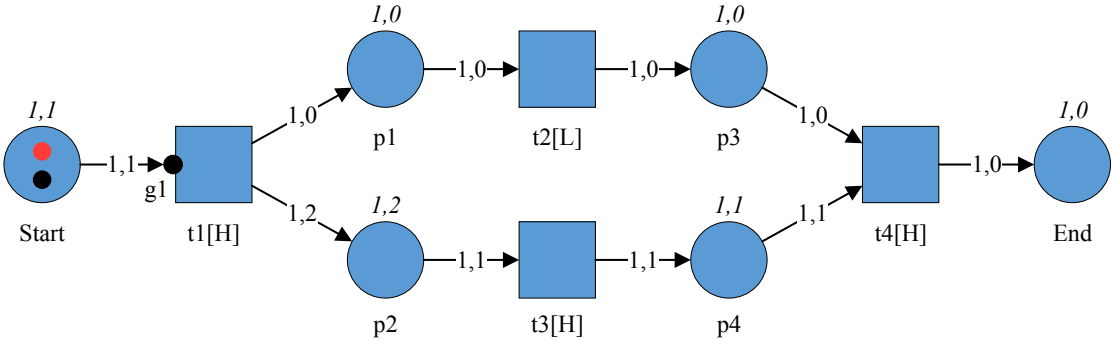


Abbildung 3.11.: Beispiel IF Net.

Der Analysekontext sei definiert durch die im Netz dargestellte Labelinformation, den Benutzern $U = \{\text{User1, User2}\}$ und der Nutzerzuweisung

$$\mathcal{S}_{\mathcal{U}} : \{(\text{t1,t4}) \rightarrow (\text{User1, User2}), \text{t2} \rightarrow \text{User2, t3} \rightarrow \text{User1}\}.$$

Im Beispiel ist die Transition $t1$ durch eine Wächterfunktion $g1$ geschützt. Diese ist durch

$$G = \{g1 : t1 \rightarrow (\text{Schreibzugriff } rot)\}$$

gegeben.

Die Zugangsfunktion sei gegeben durch

$$A = \{(t1, \text{rot}) \rightarrow \text{lesen}, (t2, \text{rot}) \rightarrow \text{schreiben}, (t3, \text{rot}) \rightarrow \text{lesen}, (t4, \text{rot}) \rightarrow \text{löschen}\}$$

Mit dieser Konfiguration kann das IFNet nicht feuern, obwohl laut den Feuerregeln aus den CPN die Transition schalten kann. Die Funktion $g1$ erlaubt das Schalten von $t1$ nur, wenn schreibend auf die rote Marke zugegriffen wird. Die Zugangsfunktion spezifiziert jedoch genau, dass der Zugriff lesend auf die Marke stattfindet. Angenommen, die Funktion G ist korrekt, handelt es sich somit entweder um einen fehlerhaften Geschäftsprozess, der nicht entsprechend den Sicherheitsbestimmungen gestaltet ist, oder die Zugangsfunktion wurde falsch spezifiziert.

3.2.4 Timed Petri Nets

Mit CPNs und IFNets sind sowohl Ressourcen als auch die Vergabe von Zugriffsrechten in Geschäftsprozessen modellier- und analysierbar. Die bisherigen Modelle abstrahieren jedoch die zeitliche Dimension von Geschäftsprozessen. Berechnungen oder Simulationen bezüglich der Zeitdauer von Prozessen und Aktivitäten oder die Nutzungsdauer von Ressourcen lassen sich somit nicht durchführen. Gerade diese Betrachtung ist sowohl zur Bewertung abstrakter als auch konkreter, monetärer Kosten und der Verfügbarkeit von Geschäftsprozessen jedoch notwendig.

Aufbauend auf klassischen Petri-Netzen wurden von Ramchandani die Timed Petri Nets (TdPNs) entwickelt. TdPNs fügen Zeit-Informationen zu den Petri-Netzen hinzu, um somit dynamische Systeme modellieren zu können. Dabei werden in diesem Modell den einzelnen Transitionen Feuerdauern zugeordnet, welche die von der Transition benötigte Zeit angibt, bis die Tokens im Nachbereich der Transition produziert sind (Ramchandani 1974).

Definition 3.7. *Ein TdPN ist ein Tupel $N = (P, T, F, W, f, m_0)$, wobei gilt:*

- *(P,T,F,W) ist das zu Grunde liegende Petri-Netz.*
- $f \mapsto T \times \mathbb{R}_+$ *: ist eine Funktion, die jeder Transition eine positive, reelle Zahl (Feuerdauer) zuordnet.* ⊣

Die Feuerregeln entsprechen denen eines Petri-Netzes. Feuert eine Transition, produziert diese jedoch das Token nicht unmittelbar. Stattdessen produziert die Transition die Tokens erst nach Ablauf der zugewiesenen Zeitschritte f. Wie in Abbildung 3.12 dargestellt, lässt sich solch eine Transition laut Ramchandi durch eine "Initialisierungs"-Transition darstellen, die durch eine Stelle p mit einer "Ausgangs"-Transition verbunden ist. Das Token bleibt dabei für f Zeitschritte im Platz p.

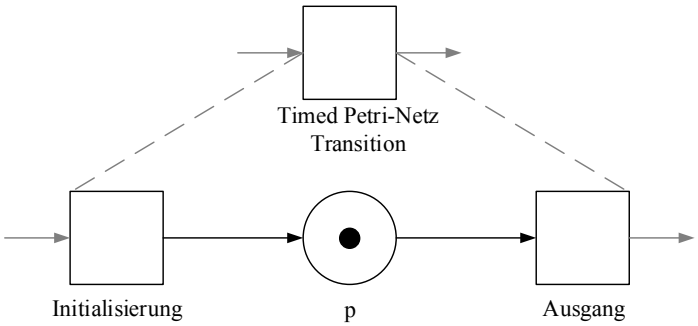


Abbildung 3.12.: Darstellung einer Transition eines Timed-Petri-Netz durch zusätzliche Transitionen

Ein TdPN besitzt zusätzlich zu seiner Markierung den aktuellen Zeitschritt, in dem es sich befindet. Feuert eine Transition t, muss die Zeit des TdPN um den Wert $f(t)$ erhöht werden damit t das Token produziert. So lange andere Transitionen feuerbereit sind, können diese Token konsumieren, bevor die Zeit erhöht wird.

In der Arbeit definiert der Autor, dass f sowohl eine positive, reelle Zahl, als auch eine Rechtecksverteilung sein kann, um somit auch die Darstellung stochastischer Prozesse zu ermöglichen. Eine als Rechtecksfunktion angegebene Zeitfunktion f gibt ein Intervall an, innerhalb dessen die von einer Transition produzierten Tokens zur Verfügung stehen.

Da Timed-Petri-Netze neben der regulären Markierung zusätzlich den Zeitpunkt, in dem sich das Netz befindet kodieren zeigen diese schnell anwachsende Markie-

rungsgraphen. Werden die Transitionen statt mit konkreten Zeitdauern mit reellen Zeitintervalle annotiert, besitzen die Netze keine diskrete Markierung mehr. Das Feuern einer solchen Transition t kann das Netz in einen beliebigen Zustand $m = ((m_o \setminus \bullet t) \cup t\bullet, d)$ mit $d \in \mathbb{R}$ überführen. Somit wird der mögliche Zustandsraum bereits durch das Feuern einer Transition unendlich. Zwar lassen sich die Zustände wieder durch eine Angabe des Intervalls zusammenfassen, doch kommt es auch unter Verwendung einer solchen Technik zu vergrößerten Markierungsgraphen (Berthomieu et al. 1983).

Beispiel

Das Beispiel in Abbildung 3.13 zeigt ein Timed Petri-Netz basierend auf dem Petri-Netz 3.6. Zusätzlich ist die Funktion f definiert als:

$$f = \{t1 \to 2, t2 \to 3, t3 \to 1, t4 \to 3\}.$$

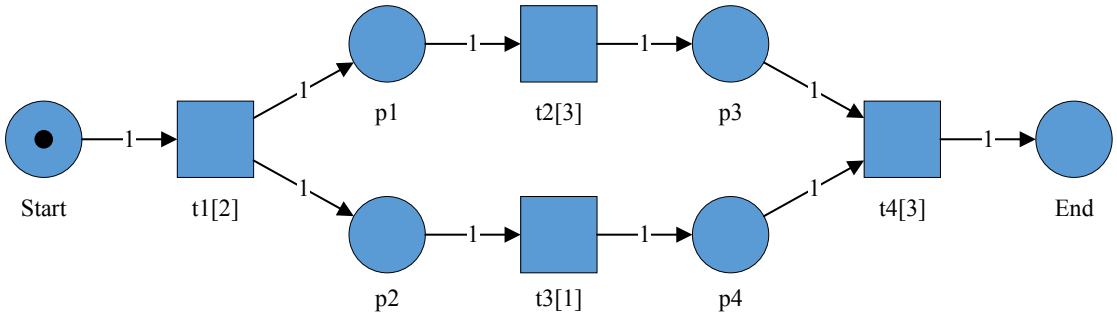


Abbildung 3.13.: Beispiel Timed Petri Net.

Nach dem Feuern von Transition $t1$ sind die Tokens in den Plätzen $p1$ und $p2$ ab dem Zeitschritt 2 verfügbar, siehe Abbildung 3.14. In diesem Beispiel sind die Zeiten, nach denen ein Token verfügbar ist, über den entsprechenden Plätzen eingetragen.

Nach dem Feuern von Transition $t1$ wird das Netz in den Zustand $(011000, 2)$ überführt. Da den Zuständen der Netze nun auch ein Zeitpunkt zugewiesen wird, vergrößert sich der Markierungsgraph, falls die Zeitfunktion f eine Rechtecksfunktion ist.

Wäre der Transition $t1$ im Beispiel statt der fixen Zeitdauer 2 eine Rechtecksfunktion mit dem Intervall (a, b) zugewiesen, dann gäbe es nach dem Feuern von $t1$ nicht

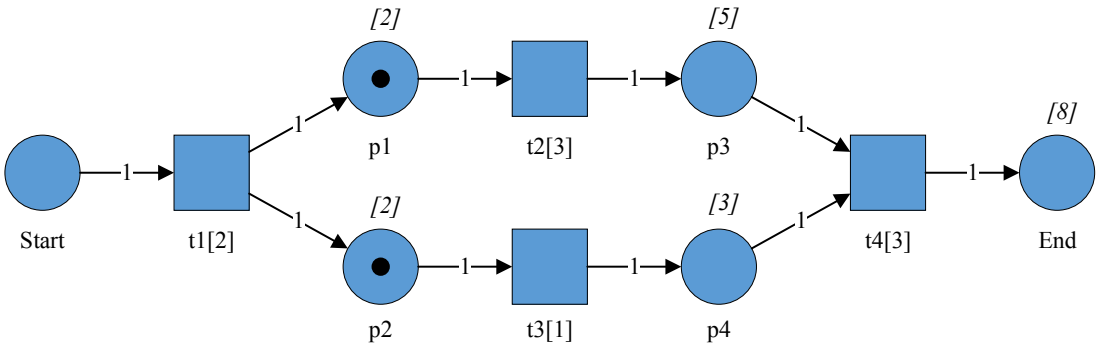


Abbildung 3.14.: Beispiel Timed Petri Net nach dem Feuern der ersten Transition.

mehr nur einen Zustand $(011000, x)$, sondern mehrere Zustände für $a \leq x \leq b$. Je größer das Intervall (a,b) ist, desto größer fächert sich der Zustandsraum auf. Die Generierung eines Markierungsgraphen kann dadurch unpraktikabel werden.

3.2.5 Time Petri Nets

Die Klasse der Time Petri Nets (TPNs) stellt eine weitere Erweiterung von Petri-Netzen dar. Ähnlich den TdPNs sind TPN eine um eine zeitliche Komponente erweiterte Klasse von Petri-Netzen (Merlin et al. 1976). Diese Erweiterung erlaubt es, jeder Transition ein nicht negatives, zeitliches Intervall $[a, b]$ anzugeben, welches dem frühesten und spätesten Feuerzeitpunkt der Transition entspricht (Aalst 1993). Dabei muss die Transition nach Ablauf des Intervalls schalten. Diese Feuerregel unterscheidet sich zu denen der Timed Petri-Netzen, da hier das Schalten einer Transition nicht erzwungen wird. Die Intervalle in einem TPN sind vom Typ INT:

Definition 3.8. $INT = \{[a, b] \in \mathbb{R}^+ \times \mathbb{R}^+ | a \leq b\}$ ⊣

Ein Wert c ist innerhalb eines Intervalls $c \in [a, b]$, wenn gilt: $a \leq c \leq b$

Im Vergleich zu TdPN verarbeitet die Transition das Token nicht, sondern belässt es in seiner Stelle bis das Intervall abgelaufen ist und die Transition schaltet. Im Gegensatz zu TPNs können Transitionen somit im zeitlichen Wettbewerb um ein Token stehen. In Time Petri Nets gilt für die mit einem Intervall markierten Transitionen:

- a gibt die minimalen Zeitschritte an, die vergehen müssen, bevor eine aktiv gewordene Transition feuern kann.
- b ist die maximale Zeit, die eine Transition aktiv sein kann, bevor sie feuern muss.
- Für das Intervall muss gelten: $0 \leq a \leq b \leq \infty$. $[a, b]$ ist somit ein gültiges Intervall nach Definition 3.8.

Die Definition der TPNs ist denen der regulären Petri-Netzen angelehnt und ist lediglich um eine Zeitintervallsfunktion erweitert:

Definition 3.9. *Ein TPN ist ein 6-Tuple $\mathcal{Z}$=(P,T,F,W,m_0,I) mit*

- *(P,T,F,W,m_0) dem darunter liegenden Petri-Netz*
- *I die Zeitintervallsfunktion $T \to \mathbb{Q}_0^+ \times (\mathbb{Q}_0^+ \cup \infty)$ für jede $t \in T$, mit $I(t) = (I_1(t), I_2(t))$, wobei gilt $I_1(t) \leq I_2(t)$.* ⊣

Die Markierung in TPNs ist um einen Zeitstempel erweitert. Somit ist jeder Markierung im Petri-Netz auch ein Zeitschritt t zugeordnet. Ist eine Transition gemäß den Feuerregeln zum Zeitschritt t aktiv, kann diese frühestens zum Zeitpunkt $t+a$ schalten. Die Transition muss spätestens bis zum Zeitpunkt $t+b$ feuern. Das Feuern der Transition selbst ist unmittelbar. Wegen der als Intervall modellierten Dauer lassen sich mit diesem Modell auch nicht deterministische Systeme und deren Zeiteigenschaften modellieren.

Time Petri Nets wurden in weiterführenden Arbeiten zur Analyse zeitlicher Systeme herangezogen und weiter erforscht (Berthomieu et al. 1983). TPN erben die gleichen Nachteile wie TdPNs: Die Generierung eines Markierungsgraphen wird unpraktikabel.

Die genannten Autoren entwickelten Methoden, um trotz der Intervallangabe einen endlichen Markierungsgraphen konstruieren zu können. Hierfür unterteilen sie die Zustände in Klassen. Eine Zustandsklasse C fasst für eine erreichbare Markierung M die Menge der Intervalle der aktiven Transitionen zusammen, unter welchen es zu dieser Markierung kommen konnte. Eine Zustandsklasse bildet somit ein Paar $C = (M, D)$ für das Intervall D. Der entstandene Graph zeigt das zusammengefasste zeitliche Verhalten des Modells. Jeder Knoten im Markierungsgraphen fasst

somit nicht mehr einen Zustand, sondern kombiniert eine mögliche Markierung des Netzes unter mehreren möglichen Zeitintervallen, in denen die Markierung vorkommen kann.

Diese Technik wurde von den Autoren zur Verifikation der zeitlichen Beschränktheit eines Kommunikationsprotokolls herangezogen und evaluiert. Die Autoren räumen selbst Nachteile ihres Analyseverfahrens ein. So ist es trotz der Nutzung von Zustandsklassen immer noch nicht garantiert, dass ein beschränkter Markierungsgraph existiert. Nur unter dieser Bedingung ist jedoch eine Erreichbarkeitsanalyse der Zustände möglich.

Zusätzlich können in TPNs die Intervall-behafteten Transitionen gegenseitig im Konflikt stehen: Gegeben seien zwei Transitionen i, j, die beide im Intervall $[a, b]$ feuern und beide Transitionen das gleiche Token konsumieren. Befindet sich nun das Netz im Zeitschritt b, müssen beide Transitionen feuern, was auf Grund der Feuerregeln der Petri-Netze jedoch nur einer Transition möglich ist. Dies führt zu einer Verletzung der erweiterten Feuerregeln der Time Petri Nets.

Beispiel

Abbildung 3.15 zeigt ein TPN mit dem zu jeder Transition zugehörigem Intervall. Die Transitionen, Plätze und Flussrelationen sind gleich mit dem Beispiel aus Abbildung 3.6. Die Intervallfunktion in diesem Beispiel lautet:

$$\{I(t1) \rightarrow [2,4], I(t2) \rightarrow [0,3], I(t3) \rightarrow [1,4], I(t4) \rightarrow [3,5]\}.$$

Die Startmarkierung ist: $m_0 = \{Start \rightarrow (1[0])\}$. Somit ist ein Token im Platz $Start$ zum Zeitpunkt 0.

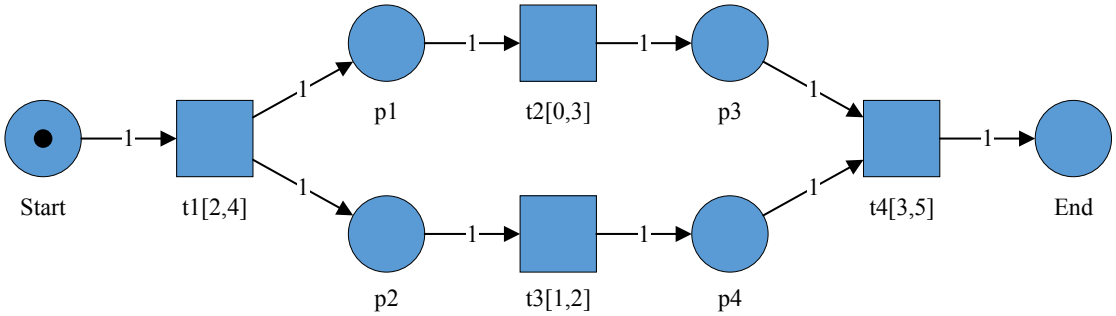


Abbildung 3.15.: Beispiel Time Petri Net.

Die Transition $t1$ kann trotz verfügbarer Token nicht feuern, da zu Beginn das Netz im Zeitschritt 0 ist, die Transition jedoch frühestens 2 Zeitschritte nach Aktivierung feuern kann. Das Feuern von $t1$ bedeutet also, dass die Zeit des Netzes um 2 bis 4 Zeitschritte weiter fortschreitet. Das Ergebnis ist in Abbildung 3.16 zu sehen. Die Intervallangabe über den Plätzen zeigt die frühesten und spätesten Zeitpunkte an, in denen ein Token in den entsprechenden Plätzen eintreffen kann. Das Netz kommt frühestens nach 6 Zeitschritten und spätestens nach 13 Zeitschritten zu einem Ende.

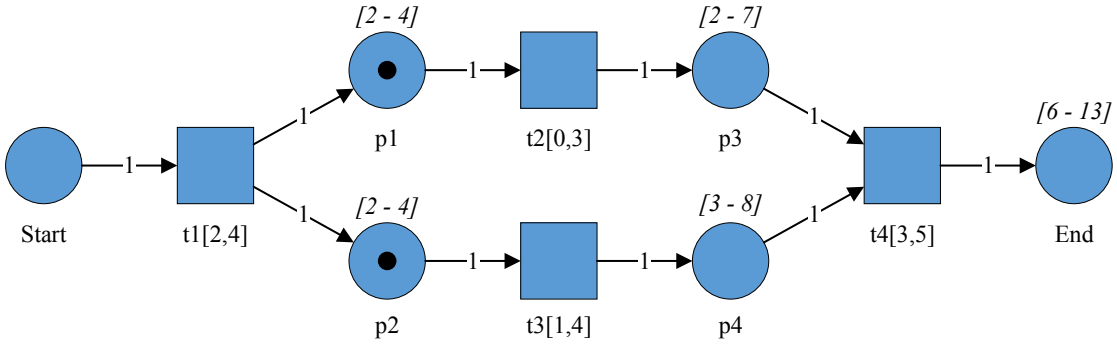


Abbildung 3.16.: Beispiel Time Petri Net nach dem Feuern der Transition $t1$.

3.2.6 Interval Timed Colored Petri Nets

In einer Studie zur Verfügbarkeit und Performanz von verteilten Web-Anwendungen von (Gillmann et al. 2000) wurde gezeigt, dass komplexe und verteilte Systeme erst durch die Kombination von CPN und TdPN vollständig abbildbar und somit analysierbar sind. Mit dieser Kombination ist es erstmals möglich, sowohl die zeitliche als auch die Ressourcen-Dimension in einem Modell zu betrachten. In *Interval timed coloured Petri nets and their analysis* wurde von Aalst[5] der Zusammenschluss beider Konzepte durch die Erweiterung von CPNs um die Funktionalität von TdPN beschrieben und dadurch ein aussagekräftiges Modell entwickelt. Das resultierende Interval Timed Colored Petri-Net (ITCPN) erweitert CPNs und heftet den einzelnen Token einen Zeitstempel an, zu welcher Zeit dieses Token produziert, also verfügbar ist. Damit lassen sich gleichzeitig Ressourcen und Zeit abbilden. ITCPNs sind definiert als:

[5]Vgl. Aalst 1993.

Definition 3.10. *Ein ITCPN ist ein 5-Tupel mit ITCPN=(Σ,P,T,C,CT,$\mathcal{F}$). Die Elemente P, T entsprechen denen eines CPN. Σ entspricht der Menge $\mathcal{C}$ der Definition von CPNs. Die Kapazitätsfunktion C bleibt erhalten.*
Die Ein- und Ausgangsfunktionen sind in einem ITCPN durch die Transitionsfunktion $\mathcal{F}$ zusammengefasst. $\mathcal{F}$ definiert die Menge und Farben der konsumierenden und produzierenden Tokens sowie ein zeitliches Intervall, in welchem die Tokens produziert werden. $\mathcal{F}$ ist analog zur Funktion f der TdPNs. Für ITCPNs gilt:

- *$\mathcal{F}$ ist die Transitionsfunktion. Wenn $t \in T$, so gilt: $F(t) \in CT_{MS} \nrightarrow (CT \times INT)_{MS}$. $A \nrightarrow B$ ist die Menge aller partiellen Funktionen (rechtseindeutige Relation $A \rightarrow_p B$) von A nach B.*
- *CT bildet die möglichen Kombinationen von Tokens in Plätzen ab: $CT = (p, v)|p \in P \wedge v \in C(p)$* ⊣

Für ITCPNs muss gelten:

- Jedem Platz $p \in P$ ist durch C die Kapazität der einzelnen Farben zugeordnet. Für jede Marke v in p muss gelten $v \in C(p)$, das heißt, die Stelle hat laut der Kapazitätsfunktion C Platz für die Marke v.
- CT ist die Menge aller farbig markierten Marken und deren Stelle. CT bildet für jede Marke auf ein Paar $< p, v >$ ab, mit der Position p und der Anzahl v.
- Eine Markierung beinhaltet für jede Marke die Stelle, in der diese Marke zu finden ist, sowie einen Zeitstempel, wann die Marke produziert wurde.
- Die Feuerregel ist durch die Funktion $F(t)$ definiert. Die Funktion bildet ab, wie viele Marken von welcher Farbe und Stelle konsumiert und wie viele Tokens welcher Farbe an welcher Stelle produziert werden. Zusätzlich gibt die Funktion das zeitliche Intervall $[a, b]$ an, in welcher Zeit die Transition feuert.
- Bei konkurrierenden Transitionen ist die Regel gegeben, dass diejenige mit der kleinsten unteren Grenze des Intervalls zuerst feuert. Ist dieser Wert gleich, kann eine beliebige Transition feuern.
- Feuert eine Transition, geschieht dies zum Zeitpunkt des zu konsumierenden Tokens mit dem höchsten Zeitstempel.

ITCPNs haben wegen den verwendeten Feuerregeln analog zu TPNs oder TdPNs schnell anwachsende Erreichbarkeitsgraphen, da die Zustände abhängig von der tatsächlichen Feuerdauer und dem Zeitpunkt des Feuerns der Transitionen abhängt. Ähnlich der Technik von (Berthomieu et al. 1983) entwickelte Van der Aalst Methoden zur Reduktion der Erreichbarkeitsgraphen, um Zeitanalysen zu ermöglichen (Aalst et al. 1995). Laut den Autoren ist in diesen reduzierten Erreichbarkeitsgraphen die notwendige Information enthalten, um das dynamische Verhalten eines Systems zu erfassen.

Die zu Grunde liegenden zeitlichen Intervalle $[a, b]$ und daraus resultierende Feuerregeln erlauben zwar eine schnelle Modellierung und ermöglichen die Reduktion eines Erreichbarkeitsgraphen, erlauben jedoch auch nur parametrisierte Aussagen über das zeitliche Verhalten. Die durch dieses Modell erzielbaren Resultate sind ebenfalls an Intervalle gebunden und können so nur einen Teil des stochastischen Verhaltens der Realität abbilden und keine Zufallsverteilung der zu erwartenden Ereignisse erzeugen.

3.2.7 Stochastic Petri Nets

Die bisher gezeigten Modelle sind nicht in der Lage, stochastische Zeitverteilungen zu ermöglichen. Stattdessen beschränken sich die bisher gezeigten Netze auf fixe oder mit Intervallen parametrisierte Zeitdauern. Die aus den Modellen erzielten Ergebnisse sind somit ebenfalls nur Angaben zeitlicher Intervalle. Die Abarbeitung von Geschäftsprozessen folgt jedoch nicht zwingend dieser Wahrscheinlichkeitsverteilung. Eine Simulation der Zeitdauern mit solchen Modellen ist daher mit einer mangelhaften Aussagekraft verbunden. Gerade im Zusammenspiel mit zusätzlichen Aktivitäten anderer Prozesse können diese ungenauen zeitlichen Angaben zu weiteren Fehlern führen. Weitere Modelle existieren für die realistische Abbildung stochastischer Prozesse. Diese ermöglichen statt der Angabe eines Intervalls die Angabe einer Verteilungsfunktion.

Die Stochastic Petri Nets (SPNs) wurden 1982 erstmals zur Analyse eingesetzt (Molloy 1982). Die Ausdrucksstärke von SPNs wurde in (Marsan et al. 1984) gezeigt.

Definition 3.11. *Ein SPN ist ein 5-Tupel SPN=(P,T,F,m_0,λ) mit:*

- *(P,T,F,m_0) dem zu Grunde liegendem Petri-Netz.*
- $\lambda = \{\lambda_0 ... \lambda_n\}$ *eine Zuordnung von exponentiell-verteilten Feuerzeiten der Transitionen im Netz.* ⊣

Stochastic Petri Net sind wegen der Gedächtnislosigkeit der verwendeten Exponentialfunktion (Farnsworth 2013) isomorph zu Markow-Ketten (Feller 2008; Howard 2012; Baier et al. 2003; Molloy 1981). SPNs erlauben aber kleinere und somit effizienter zu analysierende Modelle (Molloy 1982); vergleiche auch Abschnitt 3.1.2. Die Gedächtnislosigkeit der verwendeten Verteilung war auf Grund begrenzter Rechnerkapazitäten eine Notwendigkeit um Analysen zu ermöglichen. Die Verwendung von SPNs erlaubt es, den zu durchsuchenden Zustandsraum zu beschränken, da die Wahrscheinlichkeiten eines Überganges von einem Zustand in einen anderen unabhängig von den vergangenen Zuständen ist. Dadurch bleibt der Markierungsgraph klein. Für das Stochastische Petri-Netz bedeutet dies, dass stets ein geschlossener Term gefunden werden kann, der das zeitliche Verhalten des gesamten Petri-Netzes beschreibt.

3.3 Schwächen bestehender Petri-Netz-Modelle

Die Petri-Netz-basierten Modelle, welche sowohl zur Analyse, als auch zur Simulation nutzbar sind, wurden durch die gezeigten Entwicklung immer weiteren Bedürfnissen angepasst. Um den Anforderungen gerecht zu werden, können die erweiterten Modelle zusätzliche Details fassen. Das verfolgte Ziel ist es, detaillierte und zutreffende Prognosen über Geschäftsprozesse zu treffen, um diese innerhalb der Entwicklungsphase des BPM-Lebenszyklus nutzen zu können. So modellieren CPNs den Ressourcenverbrauch und TPNs sowie TdPNs die zeitliche Dynamik von Prozessen. Die beiden Netzerweiterungen wurden zu ITCPNs kombiniert, um die Ausdrucksstärke beider Modelle zu kombinieren, da sich erst hierdurch Prozesse in allen ihren Dimensionen betrachten lassen (Gillmann et al. 2000). Auch dieses Modell zeigt jedoch nur begrenzte Modellierungsfähigkeiten, welche den anfangs beschriebenen Anforderungen nicht genügen. Zusammengefasst sind diese Nachteile:

- Die zeitliche Komponente lässt sich in bestehenden Modellen nur ungenau als zeitliches Intervall abbilden.
- Die Abbildung von Ressourcen an Hand farblich markierter Marken erlaubt keine Abbildung geteilter Ressourcen.
- Keine hinreichende Kombination zur Betrachtung von Zeit und Ressourcen.
- Der Einfluss von Sicherheitseigenschaften auf den Prozessablauf ist nicht simulierbar.
- Bisherige Modelle lassen die simultane Simulation mehrere Prozesse mit Wechselwirkungen nicht zu.

Die Nachteile sind im Folgenden genauer betrachtet.

3.3.1 Ungenaue zeitliche Modellierungen

Für die Analyse des zeitlichen Verhaltens wurden in (Ramchandani 1974) und (Merlin et al. 1976) die Time und Timed Petri-Netze eingesetzt. Diese um Zeit erweiterten Petri-Netz Modelle erlauben jedoch keine frei wählbare Zeitverteilung. Die Zeit für das Schalten einer Transition ist entweder ein festgelegter Wert oder ein Intervall. Die damit erzielten Ergebnisse sind damit ebenfalls Intervallangaben und geben den zeitlichen Rahmen an, in dem ein Prozess frühestens oder spätestens endet. Eine statistische Betrachtung der Prozesse ist hiermit nur begrenzt möglich. Innerhalb des angegebenen Intervalls lassen sich keine Aussagen treffen, mit welcher Wahrscheinlichkeit der Prozess zu einem bestimmten Zeitpunkt endet.

Um diese sehr statische Angabe zeitlicher Parameter zu beseitigen, wurden SPNs eingeführt. Diese Klasse von Petri-Netzen erlaubt jedoch nur die Nutzung der Exponentialfunktion als ein Maß für Zeitdauern. Reale Aktivitäten in Geschäftsprozessen zeigen nicht zwingend ein solches entsprechendes Zeitverhalten. So lassen sich zum Beispiel mit dieser Verteilung Arbeitschritte, die durch einen Menschen unterstützt innerhalb eines Geschäftsprozess ablaufen, nur unzulänglich abbilden, da das menschliche Zeitverhalten stark von der Umgebung abhängig ist (Brown 1997). Bereits kleine Änderungen zeigen große zeitliche Auswirkungen durch den Menschen beim Verrichten von Aufgaben. Eine starr vorgegebene Verteilung ist somit nicht anwendbar.

3.3.2 Fehlende Modellierungsmöglichkeiten von Ressourcen

Wie mit CPNs gezeigt, lassen sich Ressourcen als farbige Token darstellen. Die Ausdrucksstärke dieser Modelle ist jedoch nicht ausreichend, um Geschäftsprozesse mit geteilten Ressourcen vollständig abzubilden, da der Typ einer Ressource in CPNs unbestimmt ist. So lässt sich mit einem CPN ohne Hilfskonstrukte wie zusätzliche Transitionen, Stellen und Marken keine Ressource modellieren, die von mehreren Aktivitäten aus gleichzeitig zugreifbar ist. Auch Ressourcen, welche durch Benutzung verschleißen und nur eine begrenzte Haltbarkeit aufweisen, sind in CPNs nicht ohne Hilfskonstrukte modellierbar.

Die Ressourcenabbildung durch CPNs trennt den Kontrollfluss, die schwarze Marke, von den Ressourcen ab. Mit diesem Modell lassen sich Transitionen modellieren, die bestimmte, unterscheidbare Ressourcen benötigen. Diese Ressourcen können von Transitionen wechselseitig konsumiert bzw. produziert werden. CPNs bieten jedoch keine Möglichkeit, Ressourcen durch mehrere Netzinstanzen hinweg zu teilen. Jede Instanz eines CPNs hat eine eigene Menge $\mathcal{C}$ an Ressourcen, welche unabhängig von anderen Instanzen ist. Somit lassen sich ohne zusätzliche Änderungen am Prozessmodell keine um Ressourcen konkurrierende Prozesse durch CPNs modellieren.

3.3.3 Keine hinreichende Kombination von Zeit und Ressourcen

Spätere Arbeiten um die Simulation von Geschäftsprozessen zeigen die Notwendigkeit kombinierter Ressourcen- und Zeitbetrachtung. Hierfür wurden die Intervall Timed Colored Petri Nets entwickelt und genutzt (Aalst 1993). Die ITCPN Modelle erben jedoch die Nachteile von CPNs, also dass Ressourcen nur unzulänglich modellierbar sind, als auch die ungenaue Angabe zeitlicher Faktoren. Diese beiden Nachteile wiegen umso schwerer in Kombination. Ressourcen werden durch Aktivitäten entsprechend ihrer Abarbeitungszeit unterschiedlich lange genutzt. Danach stehen sie anderen Aktivitäten zur Verfügung. Wenn die Abarbeitungszeit jedoch unzureichend modelliert ist, führt eine Simulation zu falschen Vorhersagen der Ressourcennutzungen. Von möglichst exakten Ergebnissen hängt jedoch ab, ob andere Aktivitäten auf die Ressourcen zugreifen können oder nicht.

3.3.4 Fehlende Simulationsmöglichkeit für Sicherheitseigenschaften

IFNets modellieren Sicherheitsanforderungen durch Wächterfunktionen, Sicherheitsdomänen und Rollenkonzepten an Prozesse und sind in der Lage Ressourcen zu modellieren. Den Einfluss von Sicherheitsanforderungen auf das Zeitverhalten eines Prozesses lässt sich allerdings wegen den fehlenden temporalen Modellierungsmöglichkeiten nicht darstellen. Zur Darstellung der negativen Effekte von Sicherheitsbestimmungen auf die Verfügbarkeit von Prozessen lassen sich IFNets nicht einsetzen.

3.3.5 Keine Simulation simultan ablaufender Prozesse

Allen Petri-Netz-basierten Modellen ist gemein, dass sie einzelne Prozesse abbilden. Das Zusammenspiel mehrerer Prozesse, also mehrerer Petri-Netze, ist nicht möglich, da die Modelle hierfür keine Schnittstellen bieten. Es lassen sich mehrere Prozesse nur durch zusätzlichen, unpraktikablen Modellierungsaufwand in einem Petri-Netz abbilden. Hierfür müssen die Geschäftsprozesse in ein einzelnes Netz überführt werden. Ressourcen, welche durch mehrere Transitionen genutzt werden, müssen manuell mit den Transitionen verbunden werden. Die Simulation eines solchen zusammengebauten Netzes zeigt wesentliche Nachteile:

- Das aus mehreren Prozessen zusammengesetzte Modell verliert seine Verständlichkeit. Da zum Beispiel gemeinsam genutzte Ressourcen mit den Transitionen aller Prozesse verbunden werden müssen, sinkt damit die Übersichtlichkeit.

- Es bedarf manueller Eingriffe um die gemeinsam genutzten Ressourcen zu verbinden.

- Die Ergebnisse aus der Simulation oder analytischen Analyse eines solchen Netzes lassen sich nicht mehr auf den einzelnen Prozess zurückverfolgen. So bietet zum Beispiel die Gesamtzeit des zusammengesetzten Netzes keine Aussage darüber, welcher Prozess tatsächlich wann fertig wurde.

Die in der Literatur verwendeten und vorgestellten Modelle erlauben weitreichende Analysen von Geschäftsprozessen und integrieren sowohl Zeit als auch Ressourcen.

Sie zeigen jedoch Schwächen gemäß den gestellten Anforderungen und können daher nicht dazu beitragen, die hier gestellten Forschungsfragen zu beantworten.

3.4 Abschließende Bemerkungen

Die existierenden Petri-Netz-basierten Modelle ermöglichten die Betrachtung verschiedener Aspekte von Geschäftsprozesse, wie Ressourcenverbrauch, Zeitbedarf oder die Einhaltung sicherheitsrelevanter Aspekte, sind jedoch nicht in der Lage, den Einfluss von Sicherheits- oder Complianceanpassungen auf das Prozessverhalten zu simulieren.

Die durch den Druck der Globalisierung entstandenen Trends wie Industrie 4.0 oder Lean-Production führen zu höheren Abhängigkeiten unter den Prozessen (Koslowski 2014). Die bisherigen Simulationsansätze beziehen sich auf einzelne Prozessmodelle, weshalb ihnen die Möglichkeit fehlt, die Abhängigkeiten von simultan ablaufenden Geschäftsprozessen zu beachten. Darauf aufbauende Simulationen, welche lediglich einzelne Prozesse betrachten, ignorieren diese Abhängigkeiten. Damit sind die somit erstellten Ergebnisse nicht zutreffend.

Dies bedeutet, dass durch das BPM zwar Änderungen an Prozessen auf Grund von Sicherheit oder Compliance durchgeführt werden, die Auswirkungen dieser Änderungen bis zum Zeitpunkt der tatsächlichen Ausführung jedoch unbekannt bleiben. Somit lassen sich unterschiedliche Umsetzung von Sicherheit und Compliance nicht im Vorhinein vergleichen. Gerade diese Fähigkeit ist jedoch von großer Bedeutung, um Stakeholderinteressen zu berücksichtigen, also die Auswirkungen verschiedener Umsetzungen auf die Verfügbarkeit der Geschäftsprozesse.

Für die sicherheitsorienterte Simulation ist die Kombination aller genannten Fähigkeiten notwendig, wie in Kapitel 2.4 gezeigt. Die bestehenden Modelle reichen daher nicht aus, um die in dieser Arbeit genannten Forschungsfragen zu beantworten. Im folgenden Kapitel wird daher das im Rahmen dieser Dissertation erarbeitete neue Modell vorgestellt, welches die Vorteile und Fähigkeiten der einzelnen Modelle kombiniert. Die Simulation dieses neuen Modells liefert Ergebnisse, welche zur Beantwortung der Forschungsfragen führen.

Kapitel 4

Resource-Timed Petri Nets

Wie im vorhergehenden Kapitel gezeigt, besitzen bestehende Prozessmodellierungssprachen Schwächen, die eine Simulation der Auswirkungen von Sicherheitsanpassungen simultaner Prozessabläufe vereiteln. Zur Beantwortung der Forschungsfragen muss es ein Verfahren jedoch ermöglichen, simultan ablaufende Geschäftsprozesse mitsamt ihres Zeit- und Ressourcenverhaltens und den Einfluss von Zugangskontrolle und Funktionstrennung oder -bindung zu simulieren. Verlangt ein Prozess beispielsweise ein Vier-Augen-Prinzip, zeigt sich kein Einfluss auf das Zeitverhalten des einzelnen Prozesses. Erst im Zusammenspiel mit mehreren Prozessen kann sich eine Wechselwirkung auf Grund der erhöhten, personellen Bindung zeigen. Die bisherigen Verfahren erlauben es jedoch nicht, dieses Verhalten zu simulieren. In diesem Kapitel werden daher die im Rahmen dieser Dissertation entwickelten Ressource-Timed Petri Nets (RTPN) vorgestellt, welche in der Lage sind, simultane Prozesse mit ihren Wechselwirkungen, deren Ressourcennutzung, und dessen Zeitverhalten zu beschreiben und zu simulieren.

Nach der Definition der RTP-Netze und deren Beschreibung folgt eine Erläuterung der Simulationstechnik der Netze. So lassen sich die RTP-Netze ähnlich gewöhnlicher Petri-Netze einzeln simulieren. Zusätzlich bieten sie jedoch die Möglichkeit der gleichzeitigen Simulation mit weiteren Netzen und deren Wechselwirkungen. Hierfür teilen sich die gemeinsam simulierten Netze einen Zustand. Somit ist gewährleistet, dass ein Netz keine Ressourcen nutzen kann, welche durch ein anderes Netz in Benutzung sind. Die Abhängigkeiten zwischen den Prozessen wird somit berücksichtigt. Netze, welche durch die Ressourcennutzung anderer Netze gestört

sind, zeigen in der Simulation das entsprechende Verhalten. Die Abhängigkeiten zwischen Prozessen werden somit aufgedeckt.

4.1 Aufbau der Resource-Timed Petri Nets

RTP-Netze integrieren die Funktionalität unterschiedlicher Petri-Netz-Variationen und erweitern diese. Ein angepasstes Simulationsverfahren der RTPN-Modelle ermöglicht den Einfluss von Zugriffskontrollen, Aufgabentrennung und -bindung sowie durch Wechselwirkungen mit anderen Prozessen aufzudecken. RTPN-Modelle kombinieren die Modellierungsfähigkeiten von:

- Colored Petri Nets zur Modellierung von Ressourcen.
- Information-Flow Nets zur Abbildung von Zugangskontrollen.
- Time Petri Nets, sowie deren Erweiterungen zur Modellierung der temporalen Aspekte einzelner Aktivitäten.

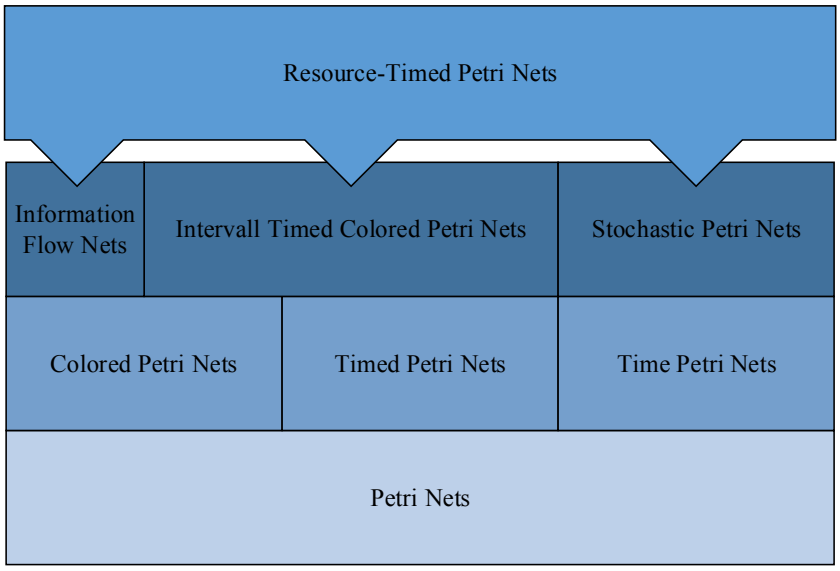


Abbildung 4.1.: Übersicht über Petri-Netze und deren Erweiterungen

Das Schichtenmodell in Abbildung 4.1 stellt dar, auf welchen Modellen RTP-Netze basieren und welche Funktionen sie vereinen: RTPNs nutzen Konzepte der IFNets, der ITCPNs sowie von SPNs. Gleichzeitig erweitern RTP-Netze die gezeigten Modelle um eine globale Markierung. Dadurch wird die Simulation nebenläufiger Prozessabläufe mit ihrer Ressourcennutzung und zeitlichen Aspekten möglich.

In Anlehnung an Timed Petri Nets, besitzen RTP-Netze einen Zeitkontext, der das zeitliche Verhalten für jede Aktivität abbildet. Dieser Zeitkontext ist jedoch nicht auf die Angabe von Intervallen oder Exponentialfunktionen limitiert. Gemäß den IFNets beinhalten RTP-Nets einen Sicherheitskontext, der es erlaubt, die vorliegenden Sicherheitskonzepte innerhalb eines Unternehmens zu modellieren. Ähnlich den Colored Petri-Nets bieten RTP-Nets die Funktionalität Ressourcen abzubilden. Dabei können die Ressourcen im Gegensatz zu den verschiedenfarbigen Tokens unterschiedliches Verhalten zeigen und die Nutzung gleicher Ressourcen aus unterschiedlichen Prozessen simuliert werden.

Hiermit erfüllen die RTPNs alle in Abschnitt 2.4 beschriebenen Anforderungen zur Simulation der Auswirkung von Änderungen der Prozessstruktur sowie der Zugangskontrolle unter der Berücksichtigung geteilter Ressourcen und Personalnutzung nebenläufiger Geschäftsprozesse.

4.2 Definition der Ressource-Timed Petri Nets

Mit den Ressource-Timed Petri Nets wird ein in dieser Arbeit neu entwickeltes Modell vorgestellt. Die RTPNs basieren wie in Abbildung 4.1 gezeigt auf Petri-Netzen. Mit diesem Modell ist es möglich, die Auswirkungen von Sicherheitsanpassungen von Prozessen auf deren Zeit-, Kosten- und Ressourcenverhalten zu simulieren. Um die Simulation nebenläufiger Prozesse zu ermöglichen, können mehrere RTP-Netze ihren Zustand untereinander teilen.

Aufbauend auf Petri-Netzen ist ein Resource-Timed Petri-Net definiert als:

Definition 4.1. *Ein RTPN ist ein Tupel*
$RTPN = ((P, T, F, W, m_0), \mathcal{S}_u, \mathfrak{R}, \mathcal{T}, d, c)$ *mit:*

- (P, T, F, W, m_0) *Die Definition des zu Grunde liegenden PT-Netzes.*
- $\mathcal{S}_\mathcal{U} : T \mapsto S$*: Teilt den Transitionen Subjekte zu.*
- $\mathfrak{R} : T \mapsto \mathfrak{r}^*$ *: Funktion, um die Verfügbarkeit von Ressourcen anzugeben.*
- $\mathcal{T} : T \mapsto \Omega$*: Weist Transitionen eine Zeitfunktion zu.*
- $d \in \mathbb{R}^+$ *: Die Zeit, bis zu welcher der Prozess geplant ist zu terminieren.*
- *c: Die aktuelle Zeit t, in der sich das Netz befindet,* $t \in \mathbb{R}_0^+$*.* ⊣

Die erweiterten Funktionen $\mathcal{S}_\mathcal{U}, \mathcal{T}$ und $\mathfrak{R}$ bilden ab, welchen Subjekten erlaubt ist, Transitionen auszuführen, wie viel Zeit das Schalten einer Transition in Anspruch nimmt und welche Ressourcen $\mathfrak{r}^*$ von einer Transition benötigt werden. Um das Modell für Analysen nutzbar zu machen, müssen diese Funktionen entsprechend ausgestaltet sein und innerhalb eines Kontexts auf die notwendigen Informationen zugreifen können.

- Der Funktion $\mathcal{S}_\mathcal{U}$ muss ein Subjektkontext hinterlegt sein, der definiert, welche Aktivitäten von welchen Subjekten ausgeführt werden dürfen. Im einfachsten Fall stellt dies eine Zugangskontrollmatrix dar.

- Dem Ressourcenkontext $\mathfrak{R}$ muss hinterlegt sein, welche Transitionen mit welchen Ressourcen arbeiten. Dies können einzelne Ressourcen oder eine Menge von Ressourcen-Sätzen sein, mit denen eine Transition arbeiten kann.

- Die Funktion $\mathcal{T}$ benötigt einen Kontext, der Informationen über das zeitliche Verhalten einzelner Aktivitäten und Ressourcen beinhaltet. Für die Funktion $\mathcal{T}$ ist eine Unterscheidung zwischen Ressourcen und Subjekten nicht von Bedeutung.

Die Ausgestaltung der Funktionen und den zugehörigen Kontexten wird in den folgenden Abschnitten erläutert.

4.2.1 Zugriffskontext $\mathcal{S}_\mathcal{U}$ der RTP Nets

Um sowohl Risiken durch Fehloperationen als auch potentielle Angriffsflächen zu minimieren, sind Unternehmen bestrebt, den Mitarbeitern durch das Rechtemanagement nur Zugriff auf die Objekte und Funktionen zu geben, die zur Vollendung ihrer Aufgabe notwendig sind (Schneider 2004). Dieses sogenannte Konzept des Least-Privilege wird durch eine Zugriffskontrolle sichergestellt, die je nach Ausprägung aus Zugangskontrollmatrix oder aus einer Rollenverteilung besteht (Bishop 2004). Diese Zugriffskontrolle als essenzieller Teil eines Prozesses muss innerhalb eines RTP-Netzen abbildbar sein, um diese korrekt zu simulieren. Die Funktion $\mathcal{S}_\mathcal{U}$, angelehnt an IF-Nets, modelliert diese Zugangsberechtigung und kann je nach Ausgestaltung verschiedene Zugriffskontrolltypen abbilden. Der Zugriffskontext $\mathcal{S}_\mathcal{U}(S, T)$ bildet somit ab, ob sich eine Transition T von einem Subjekt S feuern, also ausführen lässt. Der Evaluation der Funktion können unterschiedliche Strategien hinterlegt sein. Vorgestellt werden die Ausgestaltung durch eine Zugangskontrollmatrix, welche fein granular die Zugänge regelt und einem Role Based Access Control, welches die Zugangsrechte durch generalisierte Rollen zulässt (Bishop 2004).

Ausgestaltung als Zugangskontrollmatrix

Eine Zugangskontrollmatrix beschreibt statisch, mit welchen Rechten Subjekte auf Objekte zugreifen dürfen (Bishop 2002). Die möglichen Rechte hängen von dem zu behandelten Objekt und der Systemausprägung ab. Mögliche Rechte können zum Beispiel die Ausführung des Objektes sein, das Betrachten, Schreiben oder Betreten des Objektes. Im Falle von Geschäftsprozessen lassen sich die Aktivitäten als Objekte betrachten, für dessen Ausführung gewisse Rechte notwendig sind. Das Recht eines Subjektes eine Aktivität auszuführen, lässt sich in dem Zugriffskontext $\mathcal{S}_u$ als Matrix abbilden. Der Zugriffskontext besteht somit aus der Matrix M aus Subjekten S und Aktivitäten T, die auf 0 oder 1 abbildet.

Definition 4.2. *Der Zugriffskontext lässt sich durch eine Zugangskontrollmatrix definieren als* $AC = (S \times O) \mapsto \{0, 1\}$ *mit*

- *S: Die Menge der möglichen Subjekte*
- *O: Die Menge der Aktivitäten,* $O \subseteq T$
- $S \times O \mapsto \{0, 1\}$*: Die Zuordnung, die den Subjekten den Zugriff auf Aktivitäten erlaubt oder verweigert* ⊣

Eine 1 bedeutet, das dass Subjekt diese Aktivität ausführen darf. Im RTPN-Modell ist die entsprechende Transition also für das Subjekt gemäß den Schaltregeln feuerbar. Eine 0 bedeutet, dass das gegebene Subjekt die Transition nicht feuern, also nicht ausführen kann, selbst wenn alle anderen Bedingungen aus der Petri-Netz Definition erfüllt sind.

Wird der Zugriffskontext durch eine Zugriffskontrollmatrix gebildet, gibt die Funktion $\mathcal{S}_u(s, t)$ somit den entsprechenden Wert der Matrix M aus: $\mathcal{S}_u(s, t) = AC_{s,t}$ für ein Subjekt $s \in S$ und eine Transition $t \in O$. Existiert in AC kein Eintrag, bildet die Funktion $\mathcal{S}_\mathcal{U}$ auf 0 ab.

Ausgestaltung als Role Based Access Control

Die gezeigte Zugangskontrollmatrix ist ein einfaches Konzept zur Verteilung von Zugriffsrechten. Es bietet jedoch keine Möglichkeiten der Generalisierungen der Rechtevergabe, da jedem Subjekt jedes Recht für jedes Objekt einzeln gewährt werden muss. In Unternehmen kann solch eine Zugriffskontrollmatrix damit schnell anwachsen, da viele Subjekte und Objekte beteiligt sind. Eine Zugriffskontrollmatrix ist, bedingt durch Fluktuation von Mitarbeiter, neuen Aufgabenzuweisungen oder sich ändernden Aufgabenbereiche, sowohl arbeits- als auch wartungsintensiv. Bei jeder Änderung der Rechte oder der Subjekte muss die gesamte Matrix auf notwendige Änderungen hin geprüft werden.

Wie in der Einleitung beschrieben, steuern Unternehmen durch die organisatorische Ebene ihre Geschäftsprozesse mit Hilfe von Rollenkonzepten. Hiermit wird sowohl eine Generalisierung sichergestellt, welche die Rechtevergabe vereinfacht,

als auch die Möglichkeit gegeben, eine Aufgabentrennung und Aufgabenbindung durch die entsprechende Ausgestaltung der Rollen zu definieren.

Ein Rollenkonzept ist formal definiert als (Sandhu et al. 1996; Ferraiolo et al. 2003):

Definition 4.3. *RBAC = (S,O,R,Ro,sro,rro,sess)*

- *S: Die Menge der Subjekte*
- *O: Die Menge der existierenden Objekte* ($O \subseteq T$)
- *R: Die Menge der möglichen Zugriffsrechte auf die Objekte:* $R_{o \in O}$ *ist die Menge der Rechte auf das Objekt o.*
- *Ro: Die Menge vorhandener Rollen*
- $sro \subseteq S \times Ro$*: Die Abbildung von Subjekten zu Rollen,* $sro(s) = ro_1, ..., ro_n$*; s ist autorisiert, Rollen* $ro_1, ..., ro_n$ *anzunehmen.*
- $rro \subseteq R \times Ro$*: Die Zuweisung von Rechten zu Rollen,* $Ro \rightarrow R^o$*, mit* $r^o \in \{0, 1\} \forall r \in R$*.* r^o *ist das Recht auf Objekt o.*
- $sess$*: Die Menge der Subjekte, die aktiv in ihrer Rolle sind:* $sess \subset S \times Ro$*; Subjekte* $s_1 ... s_n$ *sind aktiv in ihrer Rolle und besitzen die zur Rolle gehörenden Rechte* ⊣

Dabei sei $Exec$ die Abbildung der aktiven Rechte eines Benutzers: $S \times R \rightarrow \{0, 1\}$ mit $exec(s, r) \rightarrow 1 \Leftrightarrow$ s hat das Recht r

Das RBAC-Modell erlaubt es, mehrere Subjekte zu definieren, die einzelne Aktivitäten ausführen dürfen. Neben der Möglichkeit einer Funktionstrennung und -bindung ergibt sich die Notwendigkeit eines solchen Rechtemodells, da es sonst zu Engpässen käme, wenn einzelne Subjekte gerade nicht zu Verfügung stehen. Statt mit den Aufgaben auf die Verfügung dieser Subjekte zu warten, sind andere Rollenmitglieder autorisiert die Tätigkeit ausführen.

Das RTPN-Modell ermöglicht die Verwendung von RBAC zur Modellierung des Zugriffskontexts. Dies stellt eine Notwendigkeit dar, um die Rechteverwaltung in Prozessarchitekturen korrekt zu simulieren. Die Abbildung 4.2 verdeutlicht die in

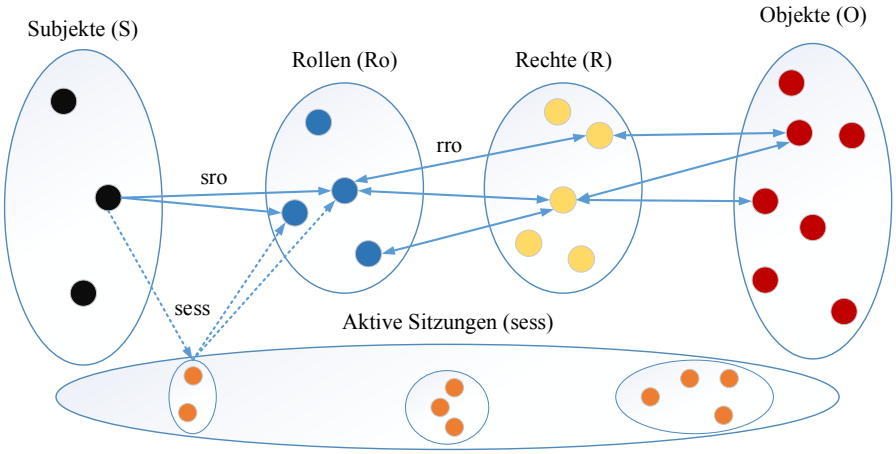


Abbildung 4.2.: Verdeutlichung der Mengen und Abbildungen einer RBAC Architektur[1]

einem Rollenkonzept existierenden Menge und Zuordnung gemäß der Definition 4.3.

Die Funktion $\mathcal{S}_\mathcal{U}$ kann als RBAC-Modell nutzen um zu ermitteln, ob ein gegebenes Subjekt eine Aktivität ausführen darf. Somit ergibt sich die Zugriffsfunktion zu:

$$\mathcal{S}_u(s,t) = \exists r \in R_t : exec(s,r)$$

Der Zugriffskontext erlaubt also die Ausführung der Aktivität t und räumt die hierfür nötigen Rechte für ein Subjekt s ein, wenn s ein Mitglied einer entsprechenden Rolle mit dem Ausführungsrecht von t ist.

Funktionstrennung in RBAC-Modellen

In einem RBAC-Modell lässt sich die Funktionstrennung (SoD) statisch festlegen. Die Zugriffsrechte der Aktivitäten, welche von verschiedenen Personen ausgeführt werden sollen, werden auf verschiedene Rollen abgebildet. Somit ist garantiert, dass jeweils nur Mitglieder einer der beiden Rollen die entsprechende Rechte zur Ausführung der Aktivität besitzt. Wird gefordert, dass eine Person nicht in beiden Rollen Mitglied sein kann, ist die Funktionstrennung durchgesetzt. Zu diesem Zweck definiert eine Menge SSD (Static Separation of Duty) die Rollen, für die ein gegenseitiger Ausschluss von Subjekten besteht:

[1] Vgl. (Ferraiolo et al. 2003)

Definition 4.4. *$SSD \subseteq R \times R$ mit*
(ro_i, ro_j): Eine gleichzeitige Mitgliedschaft eines Subjektes in Rollen ro_i und ro_j ist nicht erlaubt ($\nexists s : (ro_i, ro_j) \subseteq sro(s)$) ⊣

Für zwei Aktivitäten A und B, welche jeweils nur durch Mitglieder der Rollen ro_A und ro_B ausführbar sind, definiert die Menge SSD die Aufgabentrennung durch $SSD = \{(ro_A, ro_B)\}$. Somit kann kein Subjekt sowohl A als auch B ausführen.

4.2.2 Ressourcen-Kontext $\mathfrak{R}$

Innerhalb eines Geschäftsprozesses greifen die Aktivitäten auf verschiedene Ressourcen $\mathfrak{r}$ zu. Diese Ressourcen sind einerseits Subjekte (welche die Aktivitäten bearbeiten) sowie je nach Aktivität unterschiedliche Hilfs- und Betriebsmittel:

$$\text{Ressourcen } \mathfrak{r} : S \cup \text{Hilfs-, Betriebsmittel}$$

Die Ressourcennutzung wird in einem RTPN durch die Relation $\mathfrak{R}$ modelliert. Aktivitäten in einer Prozessarchitektur teilen sich sowohl Subjekte als auch Ressourcen. Je nach Ressourcenart können mehrere Aktivitäten gleichzeitig eine Ressource nutzen oder die Aktivitäten können nur explizit auf die Ressource zugreifen. Zusätzlich besitzt jede Ressource eine Auslastung. Diese Auslastung kann diskret oder eine kontinuierliche Angabe sein. Eine Ressource $\mathfrak{r}$ ist somit als Kombination von Typ und Auslastung definiert:

Definition 4.5. *$\mathfrak{r}$ setzt sich zusammen aus $\mathfrak{r} = (type, u)$ mit:*

$$\begin{array}{lcl} type & \in & \{explizit, geteilt\} \textit{Angabe des Ressourcentyps.} \\ u & = & \textit{Auslastung der Ressource.} \end{array}$$

Die Funktion $avail(\mathfrak{r})$ gibt die Verfügbarkeit einer Ressource $\mathfrak{r}$ zurück. Ob eine Ressource vom Typ geteilt oder explizit ist, lässt sich mit $type(\mathfrak{r})$ ermitteln.

Ein einzelner Eintrag $\mathfrak{r}$ bildet damit die Ressource selbst, deren Verfügbarkeit und Typ ab. Die Verfügbarkeit ist modelliert als das Intervall $[0, 1]$ und gibt damit an, ob die einzelne Ressource vollständig belegt (0) oder frei (1) ist. Explizit verwendete Ressourcen nehmen nur diese beiden Werte $0, 1$ an. Die Verfügbarkeit

geteilter Ressourcen kann jedoch beliebige Werte, entsprechend ihrer Auslastung, annehmen.

Der Ressourcenkontext $\mathfrak{R}$ behandelt Subjekte aus dem Rechtekontext $\mathcal{S}_U$ ebenfalls als Ressource. Dabei sind menschliche Subjekte vom Typ explizit. Automatische Agenten, die ebenfalls als Subjekt agieren können, weisen eine kontinuierliche Auslastung auf. Tatsächliche Ressourcen stellen zum Beispiel Hilfsmittel, Dienste, Datenbanken, Rohmittel und Weiteres dar.

Ressourcen-Kombinationen

Für Aktivitäten, die mehr als eine Ressource für die Abarbeitung benötigen, werden die mögliche Ressourcen-Kombinationen modelliert und als Ressourcen-Konfiguration $\mathfrak{Res}$ dargestellt:

$$\mathfrak{Res}_{t \in T} = (\mathfrak{res}_1^t, ..., \mathfrak{res}_n^t)$$

Die Ressourcen-Konfiguration fasst somit alle Ressourcen zusammen, mit welchen eine Ausführung der Aktivität t möglich ist. Aktivitäten können dabei mit einer der möglichen Ressourcen-Konfigurationen arbeiten.

Eine Aktitivät t, die beispielsweise mit zwei Konfigurationen $\mathfrak{res}_1^t = (\mathfrak{r}_1, \mathfrak{r}_2, \mathfrak{r}_3)$ und $\mathfrak{res}_2^t = (\mathfrak{r}_2, \mathfrak{r}_3, \mathfrak{r}_4)$ ausführbar ist, lässt sich folgendermaßen abbilden:

$$\mathfrak{Res}_t = (\mathfrak{res}_1^t, \mathfrak{res}_2^t)$$

$\mathfrak{Res}_{t \in T}$ beinhaltet die möglichen Ressourcen-Mengen $\mathfrak{res}_i^t$, mit der die Aktivität t ausführbar ist. Dabei ist $\mathfrak{res}_i^t$ eine Menge unterschiedlicher, zur Abarbeitung notwendiger Ressourcen für t:

$$\mathfrak{res}_n^t = (\mathfrak{r}_i, ..., \mathfrak{r}_j)$$

Verfügbarkeit von Ressourcen

Eine Ressourcen-Konfiguration $\mathfrak{res}$ ist genau dann verfügbar, wenn $avail(\mathfrak{res}) > 0$ gilt. Das bedeutet, dass alle $\mathfrak{r}$ verfügbar sind: $avail(\mathfrak{res}) := \prod_{\mathfrak{r}_i \in \mathfrak{res}} avail(\mathfrak{r}_i)$

Sobald eine der Konfigurationen in $\mathfrak{Res}_t$ verfügbar ist, gilt $avail(\mathfrak{Res}_t) := \sum_{\mathfrak{res} \in \mathfrak{Res}_t} avail(\mathfrak{res}) > 0$, womit die Aktivität ausführbar ist.

Der Funktion $\mathfrak{R}$ werden den Aktivitäten aus T die zugehörige Menge $\mathfrak{Res}$ hinterlegt:

$$\mathfrak{R} : T \rightarrow \mathfrak{Res}.$$

Eine Aktivität t ist ausführbar, wenn $\mathfrak{R}(t)$ gilt. Die Funktion wird wahr, wenn eine Ressourcenkombination $\mathfrak{res}$ existiert, für welche $avail(\mathfrak{res})$ gilt:

$$\exists \mathfrak{res} \in \mathfrak{Res}(t) : avail(\mathfrak{res})$$

In diesem Falle stehen genügend freie Ressourcen zur Abarbeitung der Transition t zur Verfügung. Gilt $\mathfrak{R}(t)$ nicht, kann die Transition t nicht schalten, selbst wenn alle anderen Bedingungen erfüllt sind.

Der Ressourcen-Kontext erlaubt es die Ressourcen-Dimension eines Prozesses abzubilden, welche entscheidend an der Verfügbarkeit eines Geschäftsprozesses beteiligt ist und Auswirkungen auf die Instanz des Prozesses zeigt.

4.2.3 Zeit-Ressourcen-Kontext $\mathcal{T}$

Um in Kombination mit der Ressourcen-Nutzung eines Prozesses Aussagen über die Verfügbarkeit, rechtzeitige Terminierung oder Kosten zu schließen, benötigen RTPN-Modelle eine Möglichkeit zur Modellierung von Aktivitätsdauern. Aktivitäten in Geschäftsprozessen zeigen nicht wiederholt das gleiche Zeitverhalten. Stattdessen folgen sie einer zufälligen Verteilung, die sich je nach ausführendem Subjekt und genutzten Ressourcen unterscheidet. Dieses Zeitverhalten spiegelt sich in der Abarbeitungszeit des Geschäftsprozesses wider und ist somit maßgeblich an der Verfügbarkeit eines Geschäftsprozesses beteiligt. Dieses zeitliche Verhalten der Aktivitäten wird durch den Zeitkontext $\mathcal{T}$ modelliert.

Für die stochastische Zeitsimulation beinhaltet $\mathcal{T}$ Informationen des zeitlichen Verhaltens der Aktivitäten unter der Berücksichtigung der verschiedenen, möglichen Ressourcen-Konfigurationen.

Für jede Aktivität $t \in T$ und möglichen Ressourcen-Satz $\mathfrak{res} \in \mathfrak{Res}$ sind die möglichen Zeitdauern als Menge positiver, reeller Zeitdauern hinterlegt.

$$\Omega_{T_{\mathfrak{Res}}} = \{d_1, ...d_n\}, |\Omega_{T_{\mathfrak{Res}}}| < \infty.$$

Diese Werte können aus einer stochastischen Verteilungsfunktion $pdf(x)$ generiert werden. Dann gilt

$$\Omega_{T_{\mathfrak{Res}}} = pdf(x).$$

Eine solche Verteilung lässt sich durch Schätzungen aus Mitarbeiterbefragungen oder durch Betrachtungen der Prozess-Logs gewinnen, wie später gezeigt.

Existiert keine Verteilungsfunktion die das zeitlicher Verhalten einer Aktivität genügend approximiert, lassen sich die Werte $d_1, ..., d_n$ direkt aus den Aktivitätsdauern der Prozess-Logs extrahieren, wie im nächsten Kapitel gezeigt wird. Dabei werden sie unverändert übernommen und in eine diskrete Verteilung überführt. Der daraus entstehende Borelsche Mengenkörper ist

$$\sigma(\Omega_{T_{\mathfrak{Res}}}) \subseteq \mathcal{P}(\Omega_{T_{\mathfrak{Res}}}).$$

Das diskrete Maß auf $\Omega_{T_{\mathfrak{Res}}}$ nach dem Vorkommen von d_i ist

$$P : \sigma(\Omega_{T_{\mathfrak{Res}}}) \to [0, 1].$$

Aus diesen Zeitdauern wird eine Zufallsvariable

$$\chi_{T,\mathfrak{Res}} : \Omega_{T_{\mathfrak{Res}}} \to \mathbb{R}$$

gebildet, die zufällig ein Element aus $\Omega_{T_{\mathfrak{Res}}}$ zieht. $\chi_{T,\mathfrak{Res}}$ modelliert die Zeitdauer, die von der Transition t mit Ressourcen $\mathfrak{res}$ bis zur Vollendung benötigt wird. Den Transitionen in einem RTPN wird diese Zufallsvariable zugeordnet. Statt eins festen Intervalls für die Zeitdauer der Aktivität, wie z.B. in TPNs, wird das zeitliche Verhalten jedes Mal aus dem Wahrscheinlichkeitsraum $\Omega_{T_{\mathfrak{Res}}}$ gezogen, wenn eine Transition feuert.

$P(t \in T, \mathfrak{res} \in \mathfrak{Res})$ sei eine Ziehung aus der Zufallsvariable $\chi_{t,\mathfrak{res}}$.

$$P(t, \mathfrak{res}) : T \times \mathfrak{Res} \to \mathbb{R}.$$

Das Feuern einer Transition führt zur Ziehung einer möglichen Zeitdauer der Transition mit den aktuellen Ressourcen. Der Zeitkontext ist definiert als:

Definition 4.6. *Der Zeitkontext $\mathcal{T}$ definiert sich als: $\mathcal{T} = (T, \mathfrak{Res}, T \times \mathfrak{Res} \to \chi)$ mit*

- *T: Der Aktivitäten bzw. Transitionen.*
- *$\mathfrak{Res}$: Den verwendeten Ressourcen.*
- *$\chi(T, \mathfrak{Res})$: Der entsprechenden Zufallsvariable für die Aktivität $t \in T$ mit Ressourcen $\mathfrak{res} \in \mathfrak{Res}$.*
- *$P(T, \mathfrak{Res})$: Der Funktion, welche aus der Zufallsvariable $\chi(T, \mathfrak{Res})$ Werte zieht.* ⊣

Somit gibt $\mathcal{T}(t, \mathfrak{res})$ die Wahrscheinlichkeitsverteilung $P(t, \mathfrak{res})$ zurück.

4.2.4 Markierung eines RTPN-Modells

Transitionen in einem RTP-Netz feuern nicht mehr wie in einem Petri-Netz augenblicklich, sondern konsumieren Tokens und produzieren diese erst nach einer Wartezeit. So lange eine Transition ein Token konsumiert, aber noch nicht produziert hat, wird die entsprechende Tätigkeit ausgeführt und die Transition ist in Arbeit. Da ein RTPN gleichzeitig aktive Aktivitäten sowohl im gleichen als auch in weiteren Prozessen modelliert, können während dieser Wartezeit weitere Transitionen feuern. Dies bedeutet, dass zusätzlich zu der Markierung m in einem RTPN der aktuelle Zeitpunkt und die Abarbeitungszeit der einzelnen Transitionen hinterlegt sein müssen.

Um den Zustand eines Netzes vollständig zu beschreiben, muss neben dem aktuellen Zeitpunkt c (in dem sich das Netz befindet), auch der Zeitpunkt, zu welchem arbeitende Transitionen t_i neue Tokens produzieren, hinterlegt sein. Zusätzlich zu der einzelnen Markierung m der Netze, stellt die Menge M die Markierung eines RTPN dar. M enthält für jede Transition t_i den Eintrag q_{t_i} vom Typ Q. Dieser Eintrag beinhaltet die Transition T, eine Menge von Ressourcen $\mathfrak{Res}$ mit welchen

die Transition gerade arbeitet und den Zeitpunkt d, ab dem die entsprechende Transition neue Tokens im Netz produziert:

$$Q = (T, \mathfrak{Res}, d).$$

Für jeden Eintrag $q_i \in M$ geben die Funktionen $T(q)$, $\mathfrak{Res}(q)$ und $C(q)$ die zugehörige Transition, den Ressourcensatz, und den Endzeitpunkt des Eintrags q zurück.

Die Markierungsmenge M ergibt sich somit zu:

$$M = \{q_{t_i} ... q_{t_j}\}, \text{mit } q_{t_i} \rightarrow Q$$

und gibt die in der Zeitspanne c bis $C(q_{t_j})$ arbeitenden Transitionen $t_i ... t_j$ an.

4.2.5 Feuerregel

Die Feuerregeln erben die RTPN von Petri-Netzen. Allerdings feuern die Transitionen innerhalb eines Zeit-Ressourcen Petri-Netzes nicht augenblicklich. Stattdessen konsumieren sie Tokens aus den vorhergehenden, direkt verbundenen Plätzen, dem Pre-Set $\bullet t$. Die Transition verarbeitet die Tokens entsprechend $ts = \mathcal{T}(t, \mathfrak{res})$ Zeitschritte. Danach produziert t die neue Tokens im Post-Set $t\bullet$, also den nachfolgenden Plätzen. Zwischen dem Konsumieren und dem Produzieren von Tokens einer Transition können weitere Transitionen innerhalb des Netzes feuern. Um die Feuerregeln zu beschreiben, wird die Relationsfunktionen $Pre\,(p, \bullet t), Pos\,(p, t\bullet) : P \times T \mapsto \mathbb{N}$ als die Anzahl der Tokens in den jeweiligen Plätzen p vor bzw. nach der Ausführung Transition $t \in T$ definiert. Dies bedeutet, es exisitieren die Kanten $(p_i, t), (p_j, t) \in F$ im Falle von $\bullet t$ oder $(t, p_i), (t, p_j) \in F$ für $t\bullet$.

Eine Transition t in einem RTPN ist feuerbereit, wenn:

- Für alle Plätze p im Pre-Set gilt, dass ausreichend Tokens vorhanden sind: $\forall p \in P, Pre(p, t) > 1$.
- Es einen erlaubten, verfügbaren Benutzer s gibt: $\exists s \in \mathfrak{R} : (\mathcal{S}_u(s, t) = 1) \wedge avail(s)$.
- Der Benutzer s in einer möglichen Ressourcen-Konfiguration aus $\mathfrak{R}$ inbegriffen ist: $\exists \mathfrak{res} \in \mathfrak{Res}_t : s \in \mathfrak{res}$.

- Die Ressourcenkonfiguration verfügbar ist: $avail(\mathfrak{res})$
- Die Transition t keinen Token bearbeitet: $\nexists q \in M : T(q) = t$.

Treffen diese Punkte zu, kann eine Transition t feuern.

4.2.6 Token Game in RTP-Netzen

Wie in Petri-Netzen lassen sich beliebige, feuerbereite Transitionen schalten. Das Schalten einer Transition führt jedoch dazu, dass Ressourcen reserviert werden, aus dem Zeitkontext eine Zeitdauer extrahiert wird und die Markierungsmenge aktualisiert wird. Im Folgenden werden die modifizierten Regeln beschrieben.

Reservieren von Ressourcen

Feuert eine Transition t, so werden die verwendeten Ressourcen $\mathfrak{res} \in \mathfrak{R}$ als belegt gesetzt. Im Falle von expliziten Ressourcen gilt somit für die Dauer der Abarbeitungszeit d $avail(res) = 0$. Für geteilte Ressourcen r innerhalb von $\mathfrak{res}$, dessen Kapazitätsgrenze noch nicht erreicht ist, gilt weiterhin $avail(r) > 0$.

Durch die Feuerregeln ist garantiert, dass ein Benutzer existiert, der sowohl verfügbar ist, als auch die Befugnis trägt, die Aktivität auszuführen.

Abarbeitungszeit

Feuert eine Transition t mit Ressourcen $\mathfrak{res}_t$, werden die Token aus dem Pre-Set $\bullet t$ entnommen. Die Transition produziert jedoch die neuen Tokens nicht unmittelbar, sondern erst nach einer Abarbeitungszeit. Diese Abarbeitungszeit d der Transition ist bestimmt durch eine Ziehung aus dem Zeitkontext: $d = \mathcal{T}(t, \mathfrak{res})$. Für diese Zeitdauer d reserviert t die Ressource. In dieser Zeitspanne sind die Ressourcen $\mathfrak{res}$ entsprechend ihrer Ausprägung in Benutzung: $avail(\mathfrak{res}_t) < 1$. Explizite Ressourcen sind nicht mehr von anderen Transitionen nutzbar: $avail(\mathfrak{res}_t) = 0$.

Aktualisierung der Markierungsmenge

Entsprechend der Abarbeitungszeit wird die Markierungsmenge aktualisiert. Diesem wird ein Eintrag q_t hinzugefügt mit $q_t = (t, \mathfrak{res}_t, d + c)$ für die gefeuerte Transition t, den verwendeten Ressourcen $\mathfrak{res}_t$ und der Zeitdauer d. Der Zeitpunkt $d + c$ gibt (ausgehende von der aktuellen Zeit c, in der sich das Netz befindet) an, wann die Abarbeitung der Transition im Netz vollendet ist.

Dynamik in RTP-Nets

Die Transitionen feuern in einem Token Game, so lange genügend Token vorhanden sind. Erst wenn keine feuerbereite Transition mehr existiert, wird die Zeit c des Netzes um den frühsten Eintrag $e \in M$ mit $\forall x : C(e) \leq C(x)$ auf den Zeitpunkt $C(e)$ hoch gesetzt. Dies erlaubt, dass ein RTP Net nicht mit diskreten Zeitschritten simuliert werden muss, sondern stets zum nächsten Ereignis aus $e \in M$ "vorgesprungen" werden kann.

Beenden einer aktiven Transition

Wird die Zeit im Netz vor gestellt, gilt $C(q_t) \geq C$ für eine Transition t, welche $\vec{M}$ entnommen wird. Der Abarbeitungszeitpunkt dieser Transition t ist damit erreicht. Es werden im Post-Set $t\bullet$ die entsprechende Anzahl von Tokens von der Transition hinterlegt. Die Ressourcen $\mathfrak{res}_t$ werden freigegeben: $avail(\mathfrak{res}_t) > 0$. Der Eintrag q_t aus der Markierungsmenge M wird entfernt. Hiermit ist das Schalten der Transition t vollendet.

Beenden eines RTP-Netz

Mit den neu produzierten Tokens können nun weitere Transitionen feuern. Ist keine Transition feuerbereit aber M belegt, wird die Zeit c des Netzes sukzessiv um die Einträge in M inkrementiert, bis ein Token das Netz wieder in einen feuerbereiten Zustand überführt. Wird die Menge leer und das Netz verbleibt ohne feuerbereite Transitionen, so ist die Ausführung des Netzes beendet.

Die Anzahl der Tokens in einem RTP-Netz kann sich somit nicht nur durch das Feuern von Transitionen verändern, sondern auch durch die zeitliche Komponente c.

4.2.7 Beispiel

Die Abbildung 4.3 zeigt ein Beispiel RTPN mit gleicher Struktur wie in Kapitel 3.2.1. Flussrelationen F, Markierung m und die Menge der Plätze P sowie Transitionen T sind identisch wie in den zuvor gezeigten Beispielen. Für die Abar-

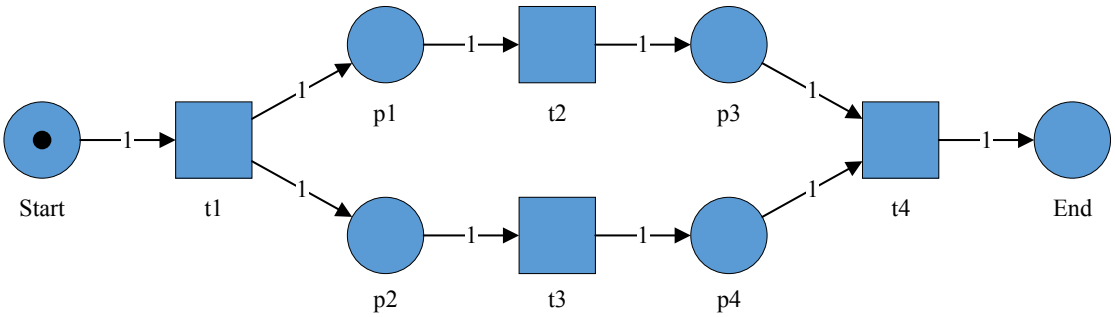


Abbildung 4.3.: Beispiel Ressource-Timed Petri-Netz.

beitung nutzt das RTP-Netz die Ressourcen und Subjekte $\{r_1, r_2, r_3, s_1, s_2\}$. Alle Ressourcen seien explizite Ressourcen und zum Zeitpunkt 0 verfügbar. Diese sind gruppiert in den Ressourcen-Kombinationen $\mathfrak{res}_1^{t_1} = \{r_1, s_1\}$, $\mathfrak{res}_2^{t_1} = \{r_2, r_3, s_1\}$, $\mathfrak{res}_1^{t_2} = \{r_2, s_1\}$, $\mathfrak{res}_1^{t_3} = \{r_1, r_3, s_1, s_2\}$ und $\mathfrak{res}_1^{t_4} = \{r_1, s_2\}$. Hiermit ist $\mathfrak{Res} = \{\mathfrak{res}_1^{t_1}, ..., \mathfrak{res}_1^{t_4}\}$. Transition $t3$ ist durch ein Vier-Augen-Prinzip geschützt. Sowohl s_1 und s_2 müssen die Aktivität ausführen.

Die Ressourcen s_1 und s_2 sind Subjekte mit folgenden Ausführungsrechten $AC = \{(s_1 \times t_1 \to 1), (s_1 \times t_2 \to 1), (s_1 \times t_3 \to 1), (s_2 \times t_3 \to 1), (s_2 \times t_4 \to 1)\}$. Alle anderen Kombination bilden auf 0 ab.

Der Zeitkontext sei: $\chi(t_1, \forall r \in \mathfrak{Res}) = \mathcal{N}(10, 4)$, $\chi(t_2, \forall r \in \mathfrak{Res}) = \mathcal{N}(8, 1)$, $\chi(t_3, \forall r \in \mathfrak{Res}) = ln\mathcal{N}(3, 0, 1)$, $\chi(t_4, \forall r \in \mathfrak{Res}) = \mathcal{N}(5, 1)$.

Die Markierungsmenge M ist zu beginn leer.

Nach dem Feuern von $t1$ mit den Ressourcen r_2, r_3, s_1 überführt sich das RTPN in folgenden Zustand (mit Zufallsziehung $\chi(t_1, \mathfrak{res}_2^{t_1}) = 12, 8$):

$$m = (000000), c = 0, M = \{(t1, \mathfrak{res}_2^{t_1}, 12, 8)\}.$$

Die Ressourcenbelegung ändert sich zu $r_2 = (explizit, 0), r_3 = (explizit, 0), s_1 = (explizit, 0)$. Da nun keine Tokens mehr vorhanden sind, muss im nächsten Schritt die Markierungsmenge abgearbeitet werden. Danach ergibt sich der Zustand $m = (011000), c = 12, 8, M = \emptyset$ und $r_2 = (explizit, 1), r_3 = (explizit, 1), s_1 = (explizit, 1)$

Von diesem Zustand aus könnten beide Transitionen $t2$ und $t3$ gleichzeitig feuern. Allerdings blockiert die gemeinsam genutzte Ressource s_1 die gleichzeitige Ausführung. Feuert nun zum Beispiel $t2$ ($\chi(t_2, \mathfrak{res}_1^{t_2}) = 7, 2$) ergibt dies den Zustand

$$m = (001000), c = 12, 8, M = \{(t2, \mathfrak{res}_1^{t_2}, 20)\}$$

und der Ressourcennutzung $r_2 = (explizit, 0), s_1 = (explizit, 0)$. Transition $t3$ kann nicht feuern und in die Markierungsmenge eingereiht werden, da eine der benötigten Ressource (s_2) nicht verfügbar ist: $avail(\mathfrak{res}_1^{t_3}) = 0$. Erst nach der Abarbeitung der Markierungsmenge ist dies möglich, wonach sich das Netz im Zustand $m = (001100), c = 20, M = \emptyset$ befindet. Dieses Beispiel zeigt, wie das Vier-Augen-Prinzip die Abarbeitung von $t3$ behindert.

4.2.8 Simulation als Token Game

Die wiederholte Ausführung des Token Games stellt die Simulation eines RTPN-Modells dar. Mit jedem wiederholten Schalten einer Transition wird eine neue Zeitdauer aus dem Zeitkontext gezogen sowie verfügbare Ressourcen zufällig benutzt und für unterschiedliche Aktivitäten reserviert. Hierdurch werden weitere Transitionen feuerbereit, da Ressourcen verfügbar oder durch einen neu entstandenen Ressourcenkonflikt an ihrer Ausführung gehindert sind. Nach einer hinreichend häufigen Wiederholung und der Betrachtung der Markierungsmenge lassen sich so die am häufigsten genutzten Ressourcen nach der Beendigung des Netzes ermitteln. Ein weiteres Ergebnis ist die Verteilung der Zeitpunkte C, also dem Ende des

Prozesses. Die gewonnenen Ergebnisse erlauben die statistische Betrachtung, wie sich ein Prozess mit den zur Verfügung stehenden Ressourcen zeitlich verhält.

4.3 Prozesssimulation

Liegt ein Prozess als RTPN vor und sind die Funktionen $\mathfrak{R}$ und $\mathcal{T}$ definiert, lässt sich das Zeitverhalten einzelner Prozesse entsprechend den Feuerregeln als Token Game simulieren:

$$SIM(rtpn) = SIM(m_0 \xrightarrow{\sigma} ... \rightarrow m_{end})) \mapsto \mathbb{R}$$

Die Simulation spielt das dem Prozess hinterlegte RTPN-Modell ausgehend von der Startmarkierung m_0 bis zu einer Endmarkierung m_{end} durch die Feuersequenz σ durch.

Die Ausgabe der Simulation umfasst mit dem Wert C des RTPNs die für die simulierte Feuersequenz σ geschätzte Prozesszeit.

Zum Erzielen aussagekräftiger Ergebnisse muss die Simulation des Netzes wiederholt werden (Henze 2013). Hierdurch werden die möglichen Ausführungspfade σ abgearbeitet und die Ergebnisse konvergieren. Die einzelnen Simulationsergebnisse für n Durchläufe lassen sich normieren und aufzeichnen. Dabei sei

$$SIM_{\varnothing,n}(rtpn) = \frac{\Sigma_n(SIM(rtpn))}{n} \mapsto \mathbb{R}$$

das durchschnittliche Ergebnis der Prozesszeit aus n Einzelsimulationen.

Das wiederholte Ausführen der Simulation, ermöglicht die Berechnung der Wahrscheinlichkeit, dass das Netz in der gegebenen Zeit d endet. Hierfür seien die Funktion $D(rtpn, x)$ folgendermaßen definiert:

$$D(rtpn, x) = \begin{cases} 1 & \text{wenn } x < \text{Deadline } d \text{ von } rtpn \\ 0 & \text{sonst} \end{cases}$$

Die Simulation $SIM_{P,n}$ sei dann:

$$SIM_{P,n}(RTPN) = \frac{\Sigma_n(D(rtpn, SIM(rtpn)))}{n} \mapsto \mathbb{R}$$

Das Ergebnis ist die durchschnittliche Wahrscheinlichkeit, dass das RTPN innerhalb seiner definierten Frist endet.

Die aus den Einzelsimulationen $SIM(M_{m0} \rightarrow ... \rightarrow M_{end})$ erhaltenen Werte lassen sich als Histogramm aufzeigen. Dies gibt an, mit welcher Wahrscheinlichkeit der Prozess zu welchem Zeitpunkt endet. Damit ist die Ausgabe aus der wiederholten Simulation sowohl die geschätzte Prozesszeit $SIM_{\varnothing}(RTPN)$ deren Verteilung als auch die Wahrscheinlichkeit $SIM_P(RTPN)$, dass der Prozess in der geplanten Zeit D endet. Als Beispiel wird der in Abbildung 4.4 gezeigte Prozess simuliert.

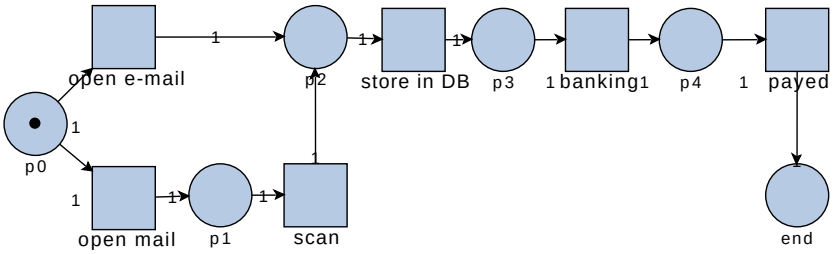


Abbildung 4.4.: Simulierter Beispielprozess

Die Ergebnisse dieser Simulation mit 250.000 Durchläufen sind in Grafik 4.5 als Histogramm abgebildet.

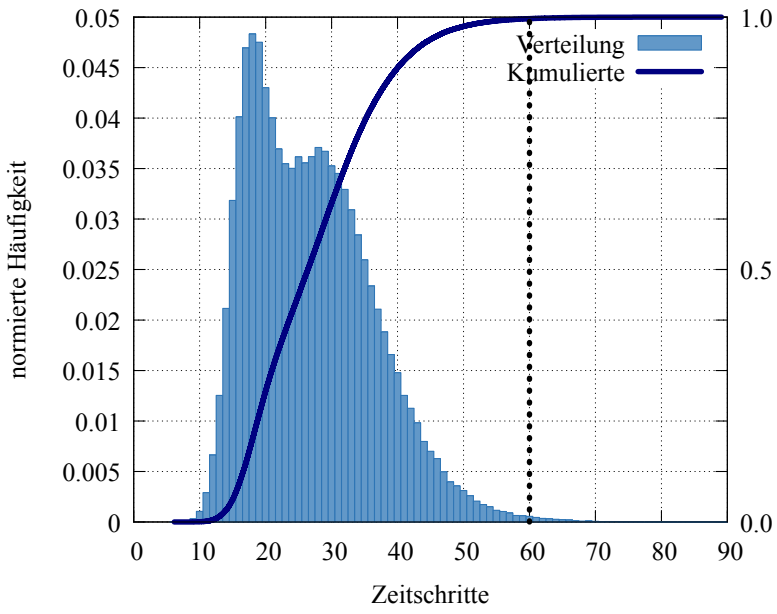


Abbildung 4.5.: Beispielergebnis der Simulation eines Geschäftsprozesses

Durch Integration der einzelnen Werte des Histogrammes lässt sich die durchschnittliche Wahrscheinlichkeit berechnen, dass der Prozess bis zum geplanten

Zeitpunkt d endet. In Abbildung 4.5 ist diese Integration als durchgehende Kurve dargestellt.

Sei für den in Abb. 4.4 simulierten Prozess die geplante Prozessdauer beispielsweise nicht mehr als 60 Zeitschritten (gestrichelte Linie in Abb. 4.5), so zeigt er eine Erfolgswahrscheinlichkeit von knapp unter 100 %. Nur sehr wenige Einzelergebnisse liegen außerhalb der Zeitspanne dieser 60 Zeitschritten, welche als gepunktete Linie dargestellt ist. Weiterhin ist erkennbar, dass der Prozess zwei lokale Maxima aufweist. Die Wahrscheinlichkeit, dass der Prozess endet ist an diesen Punkten (18. sowie 28. Zeitschritt) am höchsten. Damit sind die Ergebnisse dieser Simulation eines einzelnen Prozesses die Wahrscheinlichkeit des Prozessendes, als auch, durch Integration der Einzelwerte bis zum geplanten Prozessende, die Erfolgswahrscheinlichkeit.

4.4 Verfahren

Die Simulation spielt das dem Prozess hinterlegte RTPN ab. Ausgehend von der Startmarkierung m_0 werden die feuerbereiten Transitionen zufällig gewählt und gefeuert, was der Ausführung einer Aktivität entspricht. Das Token wird jedoch nicht direkt in das Post-Set der Transition gelegt, sondern der Zeitkontext wird befragt, wann der Token wieder zur Verfügung steht. So lange ist der Token durch die Transition in Bearbeitung und nicht im RTP-Netz verfügbar.

Die Simulation erfolgt schrittweise gemäß den Feuerregeln. So lange wenigstens eine Transition feuerbereit bleibt, ändert sich die Zeit c des Netzes nicht. Erst wenn keine feuerbereite Transition mehr existiert, muss c um den kleinsten Wert in $\vec{M}$ erhöht werden. Bis sich das Netz in einem beendeten Zustand befindet, werden weitere feuerbereite Transitionen zufällig gewählt und geschaltet. Der Algorithmus 4.1 stellt dieses Simulationsverfahren als Pseudocode dar.

4.5 Simulation nebenläufiger Prozesse

Unternehmensabläufe bestehen nicht aus einem einzelnen Prozess, sondern aus mehreren, simultan aktiver Prozesse mit unterschiedlichen Laufzeiten, Ressourcenallokationen und Aktivitäten. Eine Simulation eines einzelnen Prozesses (wie

Algorithmus 4.1 : Simulation eines RTP-Netzes

Eingabe : Das zu simulierende RTP-Netz net:
Ausgabe : Ergebniss der Simulation; benötigte Zeitschritte c: Clock

```
// Zufällige feuerbereite Transition oder Eintrag in
Warteschlange vorhanden
```

1 **while** *(*$\exists t \in$ net *mit t feuerbereit) oder (*$\vec{M} \neq \emptyset$*)* **do**
2 **if** $t \neq$ *null* **then**
3 $\mathfrak{res} \leftarrow \mathfrak{R}(t)$; `// blockiere Ressource für` t `in` $\mathfrak{R}$
4 $d \leftarrow \mathcal{T}(t, \mathfrak{res}) +$ Clock ; `// Simuliere Endzeitpunkt der Transition`
5 Entferne Tokens $\bullet t$ aus net;
6 $\vec{M} \leftarrow \vec{M} \cup (d, t, \mathfrak{res})$; `// Vollendungszeitpunkt einreihen`
7 **else**
8 $(d, t, \mathfrak{res}) \leftarrow \vec{M}_0$; `// Hole erstes Element aus` $\vec{M}$
9 $\mathfrak{R} \leftarrow \mathfrak{R} \cup \mathfrak{res}$; `// Gebe verwendete Ressource frei`
10 Erstelle Tokens $t\bullet$ in net;
11 Clock $\leftarrow d$; `// Setze neue Netzzeit`
12 **end**
13 **end**
14 **return** Clock

in Alg. 4.1) ermöglicht keine Aussagen über das Gesamtverhalten einer solchen Architektur, sondern liefert nur Ergebnisse für diesen einen simulierten Prozess, ohne externe Störungen zu berücksichtigen. Um solch eine Prozessarchitektur korrekt abzubilden, muss das RTPN-Modell seinen Zustand sowie den Ressourcenkontext mit anderen RTPN-Modellen teilen.

4.5.1 Simulation einer Prozessarchitektur

Für die Simulation einer Prozessarchitektur werden die einzelnen RTPN-Modelle in einem Tupel PA zusammengefasst, welcher die zugehörigen Prozesse, die Sicherheitskonfiguration, als auch die Ressourcen kombiniert:

Definition 4.7. *$PA = (P, \mathcal{S}_U, \mathfrak{R})$ mit:*

- *$P = \{rtpn_1, ..., rtpn_n\}$ mit den zu simulierenden Prozessen $rtpn_i$.*
- *$\mathcal{S}_U$ dem unter den Netzen geteilten Zugriffskontext.*
- *$\mathfrak{R}$ dem unter den Netzen geteilten Ressourcenkontext.* ⊣

Die Netze $rtpn_i$ können einen Zeitkontext $\mathcal{T}$ teilen oder jeweils einen eigenen besitzen.

Die Funktion $SIM(PA) \rightarrow [0...1]$ gibt das Ergebnis der Simulation der Prozessarchitektur mit den gewünschten Parametern wieder.

Dabei ist das Ergebnis der Simulation einer Prozessarchitektur der Durchschnitt der Ergebnisse der errechneten Erfolgswahrscheinlichkeiten von $\{rtpn_1, ..., rtpn_n\}$ in P:

$$SIM_{P,n}(PA) = \sum_{rtpn_i \in P} \frac{SIM_{P,n}(rtpn_i)}{|P|} \tag{4.1}$$

Die einzelnen Simulationen starten nicht sequentiell, sondern parallel, wie in Algorithmus 4.2 beschrieben. Hierdurch teilen die einzelnen Netze $rtpn_1, ..., rtpn_n$ ihren Zustand und beeinflussen sich in ihrer Ressourcenallokation gegenseitig.

Die Simulation eines einzelnen Netzes (wie im Algorithmus 4.1 gezeigt), bleibt mit leichten Änderungen erhalten. Diese in Algorithmus 4.2 gezeigten Änderungen, ermöglicht durch die nebenläufige Abarbeitung der Netze die Ressourcenabhängigkeit der Netze zu simulieren. Die einzelnen Netze teilen sich sowohl den Ressourcenkontext, als auch ihre Markierungsmenge M. Eine Ressource, die so von einem Netz reserviert wurde, ist somit für die Ausführungszeit der entsprechenden Transition auch für andere Netze nicht mehr (oder nur noch mit eingeschränkter Kapazität) verfügbar.

Das in Algorithmus 4.2 gezeigte Verfahren besteht aus folgenden Schritten:

- Zeile 1: Das Verfahren wählt aus der Menge der zu simulierenden Netzen ein Netz aus, für welches dann eine zufällige, feuerbereite Transition geschaltet wird.

- Zeile 3: Die gefeuerte Transition blockiert im geteilten Ressourcen-Kontext $\mathfrak{R}$ die benötigten Ressourcen.
- Zeile 4: Der Zeitkontext $\mathcal{T}$ bestimmt, wie lange die Ressourcen von der Transition t blockiert sind.
- Zeile 6: Die Kombination aus gefeuerter Transition t, verwendeten Ressourcen $\mathfrak{res}$ und simuliertem Endzeitpunkt $\text{CLOCK}_i + d$ wird in die geteilte Markierungsmenge M eingereiht.

Danach wird das Verfahren wiederholt. Ein neues, zufälliges Netz wird aus der Prozessarchitektur gewählt und darin eine feuerbereite Transition gefeuert. Falls keine feuerbereite Transition mehr existiert, folgt die Behandlung der Markierungsmenge M.

- Zeile 8: Das Verfahren wählt die am frühesten endende Transition t aus M.
- Zeile 9: Die von t verwendeten Ressourcen werden freigegeben.
- Zeile 10: Entsprechend der Flussrelation innerhalb des zugehörigen Netzes produziert das Verfahren die Tokens für $t\bullet$.
- Zeile 11: Alle noch nicht beendeten Netze werden auf den entsprechenden Zeitpunkt gelegt.

Die Abarbeitung der Markierungsmenge wird nur betreten, wenn keines der Netze mehr feuerbereit war. Trifft dieser Fall ein, so bedeutet dies, dass die Netze auf die Abarbeitung der nächsten Aktivität warten müssen, bis notwendige Ressourcen frei sind oder durch neu produzierte Tokens wiederum Transitionen feuerbereit werden. Die Zeit dieser wartenden Netze wird daher auf den Zeitpunkt des nächsten Ereignisses hoch gesetzt.

Die Abarbeitung von Algorithmus 4.2 ist in Abbildung 4.6 grafisch dargestellt. Es wird ein zufälliges, zu simulierendes Netz gewählt. Für dieses Netz wird, wenn möglich eine Transition gefeuert und in M eingereiht. Ist kein Netz feuerbereit, wird die frühste Transition innerhalb von M abgearbeitet. Diese Abarbeitung wiederholt sich bis zur Vollendung aller Netze.

Die Simulation eines Netzes zeigt somit durch die geteilte Zustandsmenge (der Markierungsmenge) Auswirkungen auf die gleichzeitige Simulationen anderer Netze. Eine Simulation wechselseitiger Auswirkungen wurde von bisherigen Arbeiten

Algorithmus 4.2 : Simulation einer Prozessarchitektur mit multiplen, simultan laufenden RTP-Netze

Eingabe : Die Menge der zu simulierenden RTP-Netze NET
Geteilte Warteschlange (Markierung) M
Ausgabe : Ergebniss der Simulation pro Netz CLOCK
`// Zufällige feuerbereite Transition aus zufälligem Netz vorhanden oder Warteschlange nicht leer`

```
1  while (∃i, ∃t mit t ∈ NET_i : t Feuerbereit) oder (∃e ∈ M) do
2      if t ≠ null then
3          res ← ℜ(t) ; // blockiere Ressource für t im Ressourcenkontext
4          d ← 𝒯(t, res) + CLOCK_i ;   // Simuliere Endzeitpkt d. Aktivität
5          Entferne Tokens enstprechend der Flussrelationen in NET_i ;
6          M ← M ∪ (d, t, res) ;          // Vollendungszeitpunkt für Netz i
7      else
8          (d, t, res) ← M_0;                    // Hole erstes Element aus M
9          ℜ ← ℜ ∪ res ;                  // Gebe verwendete Ressource frei
10         Füge Tokens in t• entsprechend der Flussrelationen F in NET_i hinzu;
11         for i ← 0 to |NET| do
12             if NET_i nicht beendet then CLOCK_i ← d ;        // Setze neue
               Netzzeit
13         end
14     end
15 end
16 return CLOCK
```

nicht unternommen und war mit bestehenden Petri-Netz-Modellen nicht möglich, da diese wegen fehlenden Schnittstellen nur einzelne Prozesse modellierbar machten. Mit dem in dieser Arbeit eingeführtem RTPN-Modell ist die Modellierung der wechselseitigen Beeinflussung durch Ressourcen erstmals möglich gemacht worden.

4.5.2 Simulation mit Gewichtung der Prozesse

Das Endergebnis der Simulation zeigt einen Durchschnittswert aller individuellen Simulationsergebnisse. Die Prozesse fließen jeweils zu gleichen Teilen in das Ergebnis mit dem reziproken Wert $\frac{1}{|P|}$ ein.

Einzelne Prozesse in einer Prozessarchitektur besitzen jedoch unterschiedliche Auswirkungen auf das Gesamtergebnis des Unternehmens, da sie unterschiedliche Wertschöpfungen innehaben oder einen Kernprozess darstellen

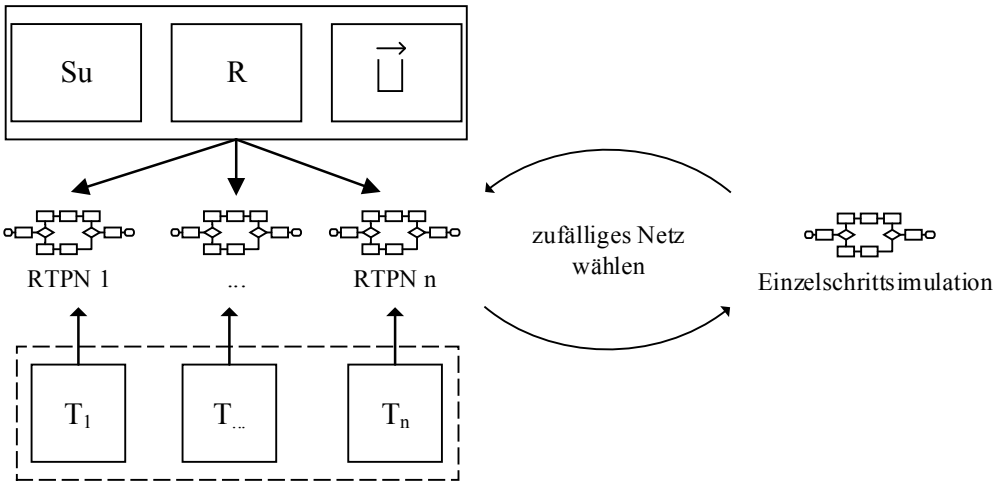


Abbildung 4.6.: Simulation einer Prozessarchitektur

(Finkeißen 1999; Becker et al. 2013). Zur Beurteilung, wie eine Prozessarchitektur auf Anpassungen und Veränderungen reagiert, ist es notwendig, diese Wertschöpfung durch eine Gewichtung der Einzelprozesse abzubilden.

So wiegt zum Beispiel der schnelle Transport einer Ware unter Einbehaltung der Kühlkette oder die Montage an einem Fließband, dessen Ausfall große monetäre Verluste zur Folge hätte, wichtiger als aufschiebbare Prozesse. Das Ergebnis aufschiebbarer Prozesse kann zum Beispiel zeitinvariant sein oder geringeren Einfluss auf das gesamte Unternehmen haben. Urlaubsanträge müssen zwar bearbeitet werden, lassen sich jedoch aufschieben.

Das Modell einer Prozessarchitektur aus Definition 4.7 lässt sich durch einen zusätzlichen Parameter G, der die Gewichtung der Geschäftsprozesse darstellt, erweitern. Mit diesem Parameter G lassen sich die Prozesse gemäß ihrer Wertschöpfung oder Wichtigkeit einordnen:

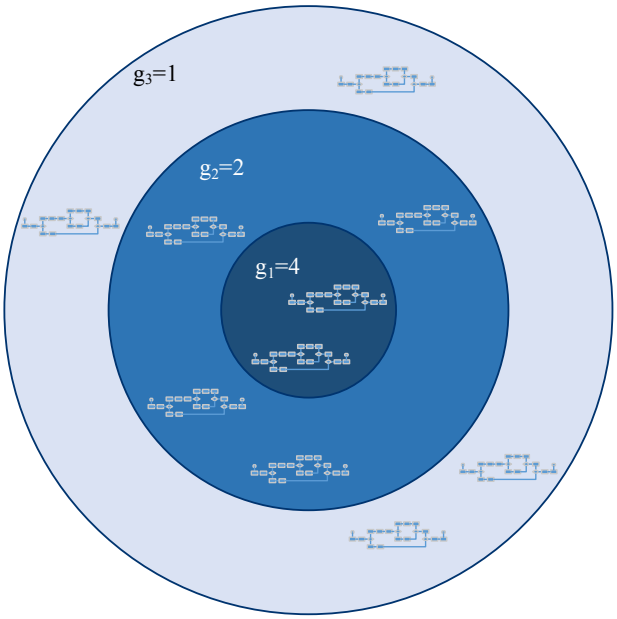


Abbildung 4.7.: Prozessarchitektur mit beispielhafter Gewichtung der Prozesse.

Definition 4.8. *$GPA = (P, G, \mathcal{S}_U, \mathfrak{R})$ mit:*

- *$P = \{rtpn_1, ..., rtpn_n\}$ mit den zu simulierenden Prozessen $rtpn_i$.*
- *$G = \{g_1, ..., g_n\}$ mit den Gewichtungen g_1 für $rtpn_i$.*
- *$\mathcal{S}_U$ einem Zugriffskontext, den alle Netze teilen.*
- *$\mathfrak{R}$ einem Ressourcenkontext, den alle Netze teilen.* ⊣

Die einzelnen Netze $rtpn_i$ können einen Zeitkontext $\mathcal{T}$ teilen oder jeweils einen eigenen besitzen.

Zur Veranschaulichung wird diese Einteilung in Abbildung 4.7 durch unterschiedliche Ringe mit unterschiedlichem Gewichtungsfaktor ($g_1 = 4, g_2 = 2, g_3 = 1$), nach außen hin abfallend, angedeutet. Sie beinhalten jeweils Prozesse, welche der entsprechenden Gewichtung zugeordnet sind. Die inneren Prozesse besitzen die höchste Gewichtung und tragen mit größerem Einfluss der Wertschöpfung bei. Prozesse der äußeren Ringe sollten die inneren Prozesse in ihrer Abarbeitung nicht negativ beeinflussen.

Die Simulationsergebnisse der einzelnen Prozesse lassen sich anhand des Normierungswertes $\frac{g_i}{\sum G}$ entsprechend nach G gewichten. Statt die Erfolgswahrscheinlich-

keiten der einzelnen Prozesse zu mitteln, lässt sich nun der Erwartungswert der Prozessarchitektur bilden, siehe (Gabler et al. 2010).

Das Gesamtergebnis ist somit der Erwartungswert der Architektur:

$$SIM_{P,n}(GPA) = \frac{\sum\limits_{rtpn_i \in P} SIM_{P,n}(rtpn_i) \cdot g_i}{\sum G}.$$

Hiermit schließt das Gesamtergebnis der Simulation einer Prozessarchitektur den Einfluss der Prozesse auf die Wertschöpfung entsprechend den Gewichtungen.

4.6 Optimierung einer Prozessarchitektur

Die Simulation einer gewichteten Prozessarchitektur GPA ist die wiederholte Ausführung von Einzelsimulationen der Prozesse: $SIM_P(GPA)$. Dabei entstehen pro Simulationsdurchlauf unterschiedliche Feuersequenzen σ, welche in der Zustandsmenge M festgehalten werden (siehe Zeile 8 aus Algorithmus 4.2). Für jeden Simulationsdurchlauf lässt sich somit für die einzelnen Feuersequenzen die Erfolgsrate festhalten:

$$\sigma_i \mapsto SIM_P^i.$$

Dabei ist SIM_P^i das zu σ_i zugehörige Simulationsergebnis aus $SIM_{P,i}(GPA)$.

Gleiche Feuersequenzen lassen sich in Klassen zusammenfassen. Für eine Klasse T, welche $|T|$ viele Feuersequenzen beinhaltet, wird der Durchschnitt der Simulationsergebnisse gebildet:

$$\sigma_T \mapsto \frac{SIM_P^i}{|T|} \; \forall i : \sigma_i \in T$$

Die Klassen geben für unterschiedliche Feuersequenzen den gemittelten Erwartungswert wieder und lassen sich nach der Erfolgswahrscheinlichkeit sortieren. Die Klasse mit der größten Erfolgswahrscheinlichkeit stellt dabei ein Optimum der ermittelten Feuersequenz dar und ist das Ergebnis der Optimierung der Prozessarchitektur.

Beispiel

Angenommen es wird der bereits in Abbildung 4.4 gezeigte Prozess optimiert. Die zwei Klassen der Feuersequenzen lauten:

σ_1 =(open e-mail, store in DB, banking, payed) sowie
σ_2=(open mail, scan, store in DB, banking, payed).

Da es sich nur um einen einzelnen Prozess handelt, können die Erfolgswahrscheinlichkeiten nur 0 oder 1 pro Simulationsschritt sein. Angenommen die Simulation liefert die einzelnen Simulationsergebnisse:

$$\begin{aligned}
\text{(open e-mail, store in DB, banking, payed)} &\mapsto 1 \\
\text{(open mail,scan, store in DB, banking, payed)} &\mapsto 1 \\
\text{(open mail,scan, store in DB, banking, payed)} &\mapsto 0 \\
\text{(open e-mail, store in DB, banking, payed)} &\mapsto 1
\end{aligned}$$

Auf die beiden Klassen σ_1 und σ_2 zusammengefasst lauten die gemittelten Ergebnisse dann:

$$\begin{aligned}
\sigma_1 &\mapsto 1 \\
\sigma_2 &\mapsto \tfrac{1}{2}
\end{aligned}$$

Die Optimierung zeigt also, dass die Ausführung gemäß der Feuersequenz σ_1 eine höhere Erfolgsquote als σ_2 aufweist.

In diesem Beispiel wurde nur die Ausführung eines einzelnen Prozesses betrachtet. Für eine Prozessarchitektur existieren im allgemeinen mehr als zwei Ausführungsklassen σ_T. Die Erfolgswahrscheinlichkeiten für einen einzelnen Simulationsschritt ist ebenfalls nicht mehr auf 0 oder 1 beschränkt, da pro Simulationsschritt die Ergebnisse aller gleichzeitig simulierten Netzen $rtpn \in P$ gemittelt werden.

4.7 Mächtigkeit von RTPN-Modellen

Die Modellierung und Simulation von Prozessarchitekturen durch RTPN-Modelle ist gegenüber Petri-Netzen um zusätzliche Fähigkeiten erweitert:

- Ein Ressourcenkontext erlaubt die Zuweisung unterschiedlicher Ressourcen und Ressourcennutzung zu Aktivitäten.

- Ein Zeitkontext erweitert die Transitionen um eine frei wählbare Funktion zur Modellierung der Wartezeit, abhängig von den tatsächlich verwendeten Ressourcen.
- Ein Zugriffskontext befähigt das Netz, Sicherheitsrichtlinien gemäß einer Zugriffskontrollmatrix oder einem RBAC-Rechtemodell abzubilden.
- Jedem Netz ist ein Zeitpunkt zugewiesen, zu dem das Netz vollendet sein soll.

Die Abbildung von Ressourcen, Zeit und Zugriffskontrolle sind jeweils bereits als Erweiterungen von Petri-Netzen modellierbar, wie zum Beispiel Colored Petri Nets, Timed Petri Nets oder Information-Flow Petri Nets. Diese bestehenden Modellerweiterungen sind jedoch nur in der Lage, stets einen Teil der Aspekte zu modellieren. Colored Petri Nets erlauben nicht die gleichzeitige Darstellung von Ressourcen und Zeit, Information-Flow Nets können kein zeitliches Verhalten abbilden und Timed Petri Nets haben keine Notation zur Modellierung von Ressourcen. Zusätzlich bietet keines der Modelle die Fähigkeit, den Zustand der gerade benutzte Ressourcen sowie die zeitliche Informationen mit weiteren Netzen zu teilen.

Wegen diesen Nachteilen lässt sich mit keinem der bisher vorhandenen Modelle eine Simulation von Prozessarchitekturen durchführen. RTP-Netze kombinieren die Möglichkeit, Ressourcen abzubilden, die Fähigkeit Feuerdauern an Transitionen zu heften und die Möglichkeit, Subjekte mit Zugriffskontrolle zu modellieren. Hiermit können RTPNs die Auswirkung sicherheitsbezogener Prozessänderungen sowohl auf der organisatorischen als auch auf der Ebene der IT-Infrastruktur simulieren. Dies bedeutet, dass die Auswirkungen von Änderungen der Zugriffskontrolle, des Rollenkonzepts sowie von Änderungen an der Struktur oder den Aktivitäten eines Prozesses im Zusammenspiel mit weiteren Prozessen quantifizierbar sind.

4.8 Prozess-Logs als Datenquelle der RTPN-Kontexte

Die Qualität der Datenbasis, auf welche sich die Simulation der RTP-Netze stützt, ist maßgeblich, um repräsentative Ergebnisse zu ermitteln. Die Kontexte des RTPN-Modells müssen für die Simulation einer Prozessarchitektur derart ausgestaltet sein, dass diese das Verhalten der Aktivitäten möglichst genau abbilden. Im Vergleich zu Schätzungen und Annahmen liefern reale Daten die Grundlage

belastbarerer Ergebnisse, da sie die tatsächlich im Unternehmen gelebte Prozesse abbilden, statt diese zu approximieren. Schätzungen können durch Mitarbeiter aus Ehrgeiz zu optimistisch ausfallen oder auf Grund des subjektiven Empfindens im Allgemeinen zu Fehleinschätzungen führen (Rezaie et al. 2009).

Ereignisse in einem Unternehmen, welches IT-gestützte Geschäftsprozesse einsetzt (beispielsweise Enterprise-Resource-Planning (ERP) oder Workflow Management Systeme (WfMS)), werden von diesen Systemen als Prozess-Logs protokolliert. Solche Ereignisse sind beispielsweise die Ausführung von Aktivitäten, das Auftreten von Störungen, der Lagerzustand oder das Beenden eines Prozesses. Wie zuvor beschrieben, ist das Gebiet des Process Minings (PM) eine Zusammenfassung von Methoden, um Kenntnisse aus den Prozess-Logs zu extrahieren (Aalst et al. 2011). Diese Technik erlaubt es, aus einzelnen Einträgen der Prozess-Logs Informationen über die Ausführung der Aktivitäten zu gewinnen. Dadurch wird ermöglicht, den Ressourcen- und Zeitkontext der RTPN-Modelle zu gestalten.

Log-Dateien entstehen im allgemeinen aus unterschiedlichen Teil-Systemen in unterschiedlichen Formatierungen und Formaten (Aalst 2016). Diese Prozess-Logs der unterschiedlichen Teilsysteme lassen sich durch geeignete Adapter in ein gemeinsames Datenformat (z.B. XES (Günther et al. 2014)) übertragen (Zahoransky et al. 2016; Accorsi et al. 2014) und werden daher in diesem Abschnitt allgemeingültig definiert. Es wird die Annahme getroffen, dass ein Prozess-Log Informationen über die Aktivitäten, deren Ausführung sowie den genutzten Ressourcen beinhaltet. Abschließend werden die im Rahmen dieser Dissertation entwickelten Verfahren vorgestellt, Informationen über die Zeitdauer und Ressourcennutzung der Prozess-Logs für die Simulation mit RTP-Netzen aufzubereiten und nutzbar zu machen.

4.8.1 Prozess-Logs als Datenbasis der Simulation

Einzelne Einträge in einem Prozess-Log beinhalten die Informationen über den Start, Abbruch, Wiederaufnahme oder der Beendigung eines Ausführungsschritt, also einer Aktivität eines Geschäftsprozesses. Wie in Abbildung 4.8 gezeigt, sind die einem Eintrag angehängten Werte die ausgeführte Aktivität selbst, der Bearbeiter der Aktivität und der Zeitpunkt des Eintrags und ob die Aktivität endete, startete oder pausiert wurde. Den Einträgen sind weitere Attribute angehängt, wie

beispielsweise die eingesetzten Ressourcen. Ein Prozess-Log ist die Aneinanderreihung dieser Einträge.

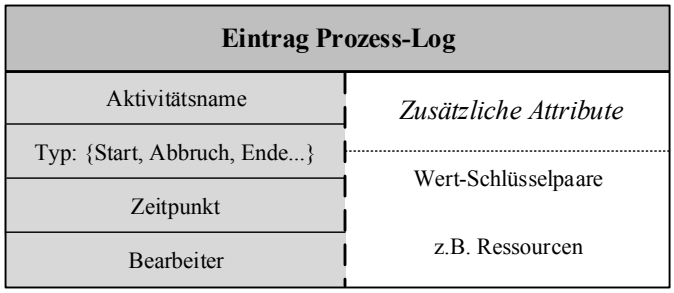


Abbildung 4.8.: Übersicht über die Attribute einzelner Einträge eines Prozess-Logs.

RTPN-Modelle erlauben es, die Einzelinformationen aus den Einträgen der Prozess-Logs zu nutzen. Verfahren, ähnlich dem PM, ermöglichen es, die zeitliche Verteilung der aus den Prozess-Logs extrahierten Aktivitätendauer zur Modellierung des Zeitkontexts zu nutzen. Weitere Felder der Prozess-Logs sind nutzbar für die Konstruktion des Ressourcenkontexts. So zeigen die einzelnen Einträge, welche Mitarbeiter die Aktivitäten bearbeitet haben und welche Ressourcen hierfür genutzt wurden. Damit ergibt sich aus der Betrachtung von Prozess-Logs ein zusammenhängendes Bild über die Prozesse. Dies beinhaltet, welche Aktivitäten von welchen Mitarbeiter mit welchen Ressourcen welches Verhalten aufweist.

Um die Verfahren zur Aufdeckung des Zeitverhaltens und der Ressourcennutzung zu erläutern, werden zuerst Prozess-Logs sowie die einzelnen Einträge formal definiert.

4.8.2 Definition von Prozess-Log

Ein Prozess-Log ist die zeitlich sortierte Abarbeitungsinformation mehrerer Instanzen eines einzelnen Geschäftsprozesses. Eine Instanz ist die tatsächliche Ausführung des Prozessmodells gestützt durch ein WfMS, ERP oder eine anderen Entität, welche die automatische Protokollierung durchführt. Einträge eines Prozess-Logs entsprechen somit den tatsächlich ausgeführten Aktivitäten des Prozesspfades eines instantiierten und somit ausgeführten Geschäftsprozesses. Die Instanzen eines Geschäftsprozesses werden im Folgenden “Case” genannt. Ein Case enthält Informationen und Details der Aktivitäten innerhalb der Instanz. Dies sind unter

anderem die Abarbeitungsreihenfolge der einzelnen Aktivitäten, deren Dauer, Bearbeiter und Ressourcennutzung. Einträge innerhalb eines Cases werden “Trace” genannt.

Abbildung 4.9 zeigt ein Klassendiagramm der einzelnen Entitäten innerhalb eines Prozess-Logs. Die ursprünglichen Klassennamen aus der Arbeit (Dongen et al. 2005) wurden beibehalten und in Klammern übersetzt. Ein Prozess-Log beinhaltet dementsprechend ein oder mehr protokollierte Prozesse mit den jeweiligen Instanzen. Jeder Instanz sind die einzelnen Aktivitäten und deren Ausführungsdetails zugeordnet.

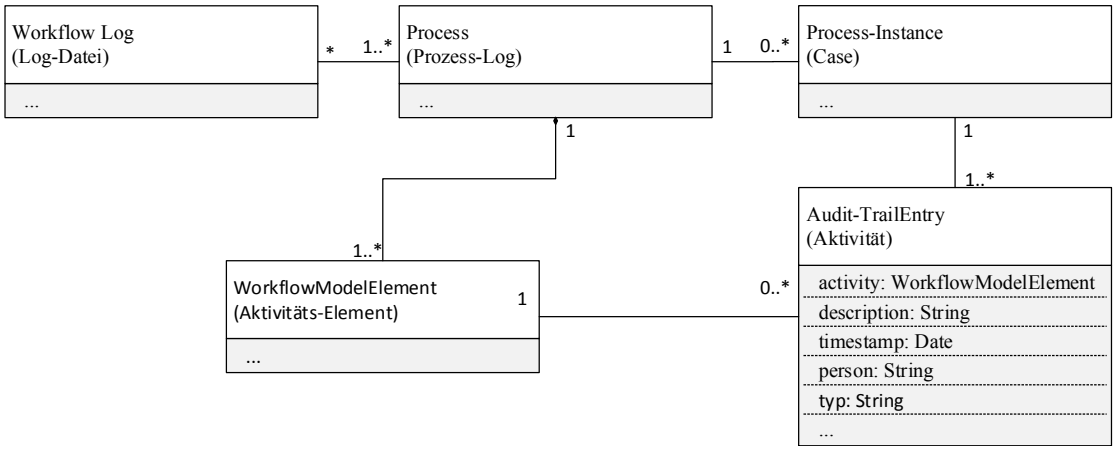


Abbildung 4.9.: Klassendiagramm einer Log-Datei[2]

Definition Case und Prozess-Log

Seien A die Menge der Aktivitäten eines Geschäftsprozesses, so ist ein Case $\sigma \in A^*$, wenn σ eine Reihe von Aktivitäten $a \in A$ enthält, die zur Abarbeitung der Prozess-Instanz geführt haben. Weiterhin sei ein Prozess-Log P definiert als eine Liste aus Cases: $P \subseteq \mathcal{P}(\sigma)$.

Zeitnotation in Prozess-Logs

Den Elementen $a \in A^*$ sind zeitliche Informationen zugeordnet. So ist jeder Aktivität a ein Typ zugeordnet, der angibt, ob die entsprechende Aktivität gestartet,

[2] nach (Dongen et al. 2005)

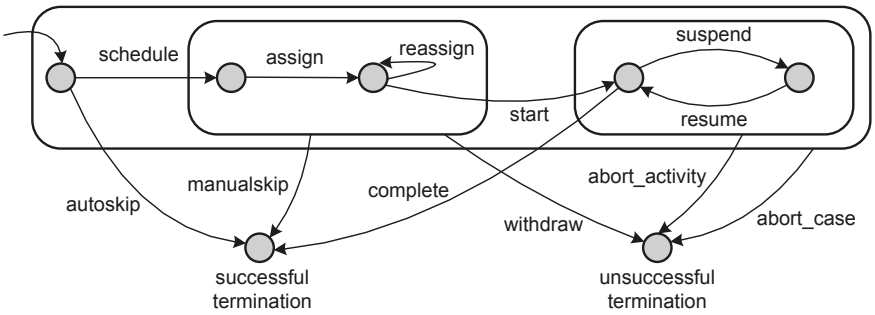


Abbildung 4.10.: Lebenszyklus einer Aktivität innerhalb eines Prozess-Logs[3].

beendet, gestoppt oder unterbrochen wurde. In (Aalst 2011) wurden die möglichen Zustände einer Aktivität modelliert und als Zustandsautomat angegeben, siehe Abbildung 4.10.

Einzelne Aktivitäten $a \in A^*$ sind wegen der Angabe des Typs mehrfach pro Case vorhanden und zwar, je nach Zustand, von welchem sie gewechselt haben. Zu jedem Eintrag ist ein Zeitpunkt hinterlegt, wann die Zustandsänderung geschah. Somit sind für erfolgreich durchgeführte Aktivitäten mindestens deren Startzeitpunkt (Änderung zu "start") und Endzeitpunkt (Änderung zu "complete") hinterlegt.

In solch einem Fall sei

$$t_s(a) \in \mathbb{R}_+$$

der Startzeitpunkt einer Aktivität sowie

$$t_e(a) \in \mathbb{R}_+$$

der Endzeitpunkt der Aktivität.

Die Dauer der Aktivität $t_d(a)$ ergibt sich dann als die Differenz beider Zeitpunkte:

$$t_d(a) = t_e(a) - t_s(a).$$

In Fällen, in denen eine Aktivität weitere Zustände durchläuft, sind weitere Definitionen der Aktivitätsdauer möglich, wie zum Beispiel der Dauer $t_w(a)$, der tatsächlich an der Aktivität erbrachten Arbeitszeit. Für diese Zeitdauer muss die Zeitspanne von "suspended" bis "resume" von der Dauer $t_d(a)$ abgezogen werden.

[3]Vergleiche (Aalst 2011).

Endet eine Aktivität mit dem Zustandsübergang “abort_case” oder “abort_activity”, so wurde die Aktivität nicht erfolgreich beendet. Damit ist keine Aussage über die Zeitdauer dieser Aktivität möglich.

Ressourcendetails einer Aktivität

Jeder Aktivität a seien zusätzliche Informationen hinterlegt, die deren Ressourcennutzung kennzeichnen. Der ausführende Bearbeiter oder das ausführende System wird als $o(a)$ hinterlegt. Die Liste der genutzten Ressourcen wird in $res(a)$ abgelegt und beinhaltet die Menge aller genutzten Ressourcen, die von a innerhalb der Zeitspanne $t_d(a)$ genutzt wurden.

Beispiel Log-Datei

Abbildung 4.11 zeigt ein Beispiel einer Log-Datei. Dabei bildet die Log-Datei zwei Instanzen eines fiktiven Bestellprozesses ab. Abgebildet sind jeweils die Aktivitäten “Bestellung aufgeben” und “Geldbetrag freigeben”. Das Beispiel zeigt nicht den vollen Prozess. Dieser könnte aus weiteren Instanzen und Aktivitäten bestehen, symbolisch durch “...” dargestellt.

Abbildungen auf Dateiformate

Es existieren verbreitete, generische Formate für Prozess-Logs, die durch unterschiedliche Werkzeuge Unterstützung erfahren und eingesetzt werden. Die gängigen Formate erfüllen die genannten Anforderungen. Das bedeutet, sie können zu einzelnen Aktivitäten das Subjekt, die genutzten Ressourcen sowie die Abarbeitungszeitpunkte speichern. Das MXML-Format, ein Meta-Modell für Prozess-Mining-Daten, wurde an der technischen Universität Eindhoven entwickelt (Dongen et al. 2005). MXML speichert zu jeder einzelnen Aktivität, welches Subjekt die Aktivität zu welchem Zeitpunkt begonnen hat und beinhaltet Möglichkeiten, zusätzliche Nutzerdaten an Prozess-Logs und darin enthaltenen Aktivitäten zu binden. Die Daten sind jeweils als Schlüssel-Werte-Paar hinterlegt. Beliebige Werte lassen sich darin abbilden, auch die genutzte Ressourcenmenge.

Der Nachfolger des MXML-Formats ist das Extensible Event Stream Format (XES) (Günther et al. 2014), ebenfalls entwickelt von der Technischen Universität

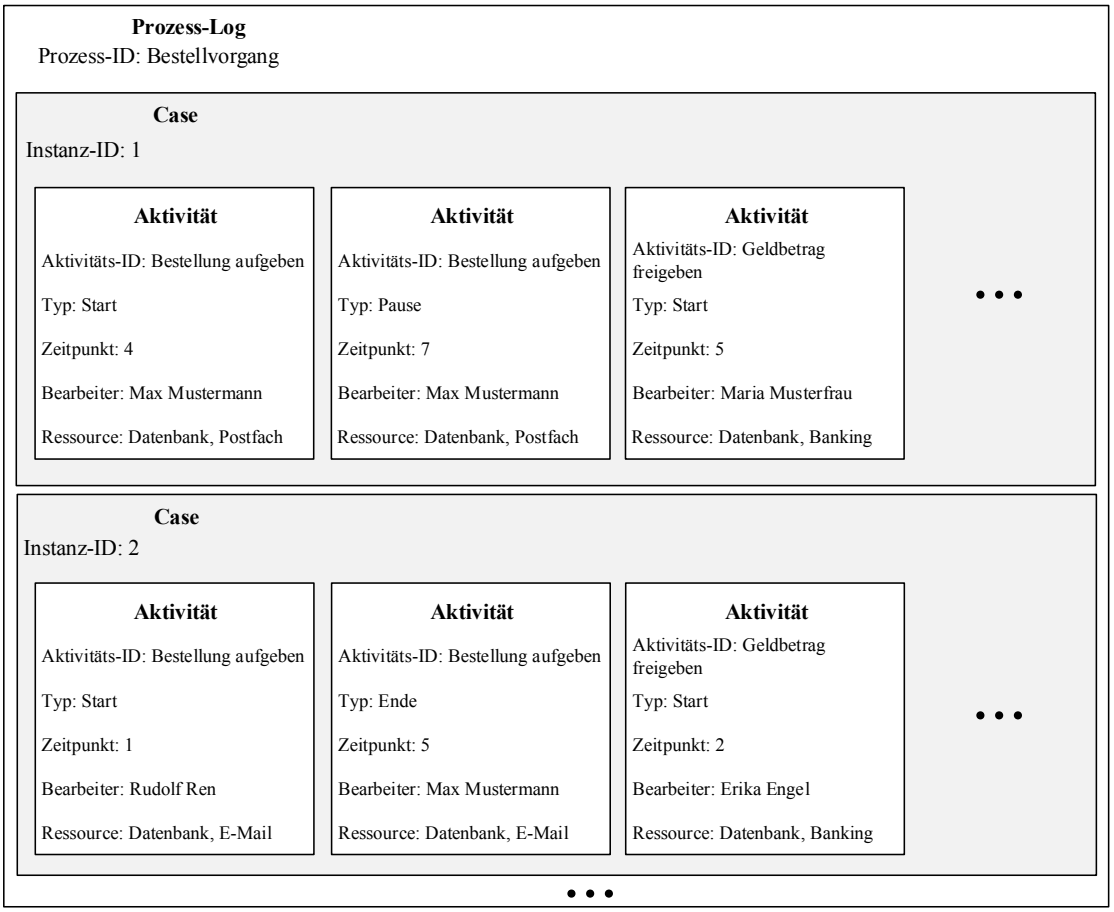


Abbildung 4.11.: Beispiel einer Log-Datei.

Eindhoven. Wie MXML entspricht XES einem XML-Format und strebt Erweiterbarkeit, Flexibilität und Ausdrucksfähigkeit an. Das XES-Format beinhaltet pro Datei einen Log mit mehreren Instanzen. Innerhalb jeder Instanz folgen Events, also die Abarbeitungsreihenfolge der einzelnen Aktivitäten. Zu den einzelnen Events lassen sich Schlüssel-Werte-Paare mit beliebigen Daten hinterlegen. Eine Erweiterung gegenüber dem MXML-Format ist die Möglichkeit, den Aktivitäten-Zyklus genauer zu beschreiben. Das Format bietet Unterstützung zu dem Lebenszyklus von Abbildung 4.10. Somit bietet dieses Format mehr Möglichkeiten der Zeitbestimmung von Aktivitäten.

4.8.3 Ressourcen-Mining

Ressourcen und Betriebsmittel stellen wichtige Kernbereiche des Unternehmens dar. Ohne passende und ausreichende Ressourcen können Mitarbeiter nicht die

Aktivitäten bearbeiten, die notwendig sind, um die Unternehmensziele zu erfüllen. Unterschiedliche Aktivitäten verlangen unterschiedliche Ressourcen, welche sich auch in ihrer Zugriffsart unterscheiden. Manche sind exklusiv, können also nur von einer Aktivität oder einer Person gleichzeitig benutzt werden, wohingegen auch geteilte Ressourcen existieren, die von mehreren Aktivitäten gleichzeitig nutzbar sind. Diese unterschiedliche Ausprägung stellt den Kern des Ressourcen-Kontexts der hier ausgearbeiteten RTPN-Modelle dar. Die Ausgestaltung dieses Kontexts ist von zentraler Bedeutung, um die Abhängigkeit zwischen Prozessen in Prozessarchitekturen zu simulieren. Nutzt ein Unternehmen ein ERP- oder WfMS, halten diese die Zusammenstellung von Ressourcen und Hilfsmittel fest. Somit stehen dem Unternehmen stets die aktuellen Informationen über Ressourcen und deren Auslastung zur Verfügung. Diese Informationen sind durch die Prozess-Logs verfügbar und lassen sich für den Ressourcen-Kontext nutzbar machen.

Verbesserungen und Neuentwicklungen von Prozessen sowie Abschätzungen der Produktivität erzwingen die Betrachtung der im Unternehmen vorhandenen Ressourcen (Russell et al. 2005), da das Verhalten von Prozessen durch die Ressourcen-Nutzung und deren Verfügbarkeit abhängt. Die in Prozessen und deren Aktivitäten verwendeten Ressourcen lassen sich als BPMN-Modell zwar definieren, doch wird diese Funktion selten umgesetzt (Grosskopf et al. 2009). Gerade in Prozess-Architekturen lassen sich die genutzten Ressourcen manuell nur mit hohem Zeitaufwand modellieren. Eine Betrachtung der Ressourcenauslastung während der Entwicklung eines Prozesses unterliegt somit Schätzungen und ungenauen Angaben. Die Log-Dateien der Prozesse hingegen beinhalten die von jeder Aktivität tatsächlich genutzten Ressourcen und stellen eine sehr detaillierte Datenbasis dar.

Um diese Nachteile zu umgehen und nicht alle Ressourcen innerhalb des RTPN-Modells manuell zu definieren, lässt sich der Ressourcen-Kontext durch die im Prozess-Log hinterlegten Informationen erstellen. Der Typ einer Ressource ist unter Verwendung einzelner Prozess-Logs jedoch nicht ermittelbar. Stehen Log-Dateien einer gesamten Architektur zur Verfügung, lassen sich Nebenläufigkeiten erkennen und diese Information in den Ressourcen-Kontext integrieren. Arbeiten unterschiedliche Aktivitäten zur selben Zeit mit gleichen Ressourcen $\mathfrak{r}$, bedeutet dies eine geteilte Ressource. Das RTPN-Simulationsmodell unterscheidet zwischen diesen Ressourcenarten.

Geplante Prozesse zeigen während der Ausführung häufig ein abweichendes Verhalten (Zahoransky et al. 2016). So ist es möglich, dass andere Ressourcen als geplant genutzt werden oder verschiedene Personen die Aufgaben unterschiedlich abarbeiten. Eine Simulation von Prozessen, basierend auf der eigentlich geplanten Prozessnutzung, würde so unweigerlich zu falschen Ergebnisse führen. Die in den Log-Dateien hinterlegten Information geben die tatsächlich genutzten Ressourcen und das tatsächliche Verhalten wieder. Log-Dateien stellen auch Krankheitsfälle, Abwesenheit oder Urlaub von Mitarbeitern, sowie den tatsächlichen Bedarf an Betriebs- und Hilfsmitteln und Lagerbestände ab. Die Daten entsprechen also nicht lediglich dem geplanten, sondern dem realen Ablauf.

Die in den Prozess-Logs vorhandenen Informationen in den Ressourcenkontext zu überführen ermöglicht eine realitätsnahe Simulation der Prozesse.

Anforderungen an Log-Dateien

Für die Extraktion der Ressourcennutzung aus Log-Dateien müssen die entsprechenden Informationen zur Aktivität a in den Prozess-Logs vorhanden sein. Dies setzt mindestens die Anwesenheit der Information über den Bearbeiter $o(a)$ einer Aktivität voraus. Dieses Feld identifiziert den Benutzer, welcher die Aktivität tatsächlich ausgeführt hat. Der Ressourcenkontext in RTPNs unterscheidet nicht zwischen Ressourcen und Subjekten. Daher kann der Ressourcenkontext bereits mit diesen Informationen erstellt werden. Mit dieser Information lässt sich simulieren, wie sich die Prozesse mit gegebenen Beschränkungen durch Sicherheitsrichtlinien wie Aufgabentrennung oder dem Vier-Augen-Prinzip verhalten.

Halten die gefundenen Log-Dateien die Informationen zur Ressourcennutzung $\mathfrak{res}(a)$ der Aktivität a bereit, lassen sich die Informationen, welche Ressourcen die Aktivität a benötigt, hinzufügen.

Ressourcen-Typ bestimmen

Die Art des Ressourcen-Typs lässt sich durch eine Betrachtung der Prozess-Logs abschätzen, wenn angenommen wird, dass die gesamte Prozessarchitektur in den Prozess-Logs erfasst ist. Dies ermöglicht die nebenläufigen Prozesse zu betrachten und somit auch konkurrierende Ressourcennutzung zu erkennen. Treten in diesen

Log-Dateien Ressourcen auf, die zu gleichen Zeitpunkten von unterschiedlichen Aktivitäten innerhalb eines oder mehrerer Prozessen genutzt werden, so müssen diese Ressourcen einen geteilten Zugriff erlauben. Wird in den Logs die zeitliche Verwendung der Ressourcen verglichen, so zeigen exklusiv genutzte Ressourcen keine Überschneidung. Das Verfahren ist in Algorithmus 4.3 dargestellt.

Algorithmus 4.3 : Ressourcentypen aus Prozess-Logs bestimmen.

Eingabe : Menge der Prozess-Logs L
Ausgabe : Feld mit Angabe des Ressourcentyps
Ressourcen (Ressource, Typ)

```
1  Typ Zeitintervall: (Startzeitpunkt start, Endzeitpunkt end);
2  Array Res ←(Ressource, Array Zeitintervalle);        // Ressourcennutzung
3  foreach P ∈ L do              // Für alle Logs der Prozessarchitektur
4      for a ∈ P do                                  // Für alle Aktivitäten
5          if a_typ ∈ {start,resume} then
               // Zeitintervall der Ressourcennutzung hinzufügen
               Res.add(res(a), Zeitintervall (t_s(a), t_e(a));
6          end
7      end
8  end
   // Für alle gefundenen Ressourcennutzungsintervalle in Res
9  foreach Ressource ∈ Res do
10     foreach Zeitintervall ∈ Res_Ressource do
11         Startzeiten ←Zeitintervall.sort(start) ;        // Nach start sortieren
12         Endzeiten ←Zeitintervall.sort(end) ;            // Nach end sortieren
13         if Startzeiten ≠ Endzeiten then
               // Überschneidung in Ressourcennutzung gefunden
14             Ressourcen ←(Ressource, geteilt);
15         else
               // Keine Überschneidung in Ressourcennutzung gefunden
16             Ressourcen ←(Ressource, exklusiv);
17         end
18     end
19 end
20 return Ressourcen
```

Dieser besteht aus folgenden Teilen:

- Zeilen 1-8: Aus den Prozess-Logs der Prozessarchitektur erstellt das Verfahren ein Feld mit den verwendeten Ressourcen und den Zeitintervallen der Nutzung.

- Zeile 11: Für jede gefundene Ressource werden die Nutzungsintervalle [Nutzungsbeginn, Nutzungsende] aufsteigend nach deren Nutzungsbeginn sortiert.
- Zeile 12: Es wird ein zweites Mal sortiert, diesmal aufsteigend nach dem Nutzungsende der Ressource.
- Zeile 13: Im Falle eines stabilen Sortierverfahren sind die beiden Listen gleich, wenn keine Überschneidungen in der Nutzung der Ressource auftreten. Treten Überschneidungen auf, bedeutet dies die Existenz zweier aufeinander folgender Intervalle $[a, b], [c, d]$ mit $b > c$. Die sortierten Listen unterscheiden sich, da die Intervalle in diesem Fall nicht monoton steigend sind. Dies heißt, die entsprechende Ressource wurde im Prozess-Log erneut genutzt, bevor sie zuvor freigegeben wurde.
- Zeile 14: Wurde eine Überschneidung der Ressource gefunden, wird diese als eine geteilte Ressource eingestuft.
- Zeile 16: Wurde für eine Ressource keine Überschneidung in der zeitlichen Nutzung gefunden, wird diese als exklusiv eingestuft.

Ausgehend von den Prozess-Logs der Prozessarchitektur ist das Ergebnis des Algorithmus 4.3 eine Liste der verwendeten Ressourcen sowie die Angabe über deren exklusiven oder geteilten Verwendungstyp.

Eine geteilte Ressource kann mindestens so viele Aktivitäten gleichzeitig bedienen, wie zeitliche Überschneidungen in den Log-Dateien gefunden wurde. Dieser Wert ließe sich zur Abschätzung der maximalen Kapazität der Ressource nutzen.

Im Beispiel aus Abbildung 4.11 ist die Ressource Postfach eine exklusiv genutzte Ressource, wohingegen mehrfach und parallel auf die Datenbank zugegriffen wird.

Ressourcennutzung der Aktivitäten bestimmen

Nachdem der Typ jeder Ressource bekannt ist, benötigt der Ressourcenkontext $\mathfrak{R}$ für seine Funktionalität die Zuordnungen von Aktivitäten zu Ressourcen. Diese Abbildung geschieht durch die Ressourcen-Menge $\mathfrak{Res}$, die für alle Aktivitäten a die verwendeten Ressourcen in $\mathfrak{Res}_a$ bereithält. Das Verfahren zum Erstellen der Ressourcen-Menge $\mathfrak{Res}$ ist in Algorithmus 4.4 dargestellt.

Hierfür wird durch die vorhandenen Log-Dateien iteriert. Für jede Aktivität a wird dabei ein Eintrag $\mathfrak{res}_i^a$ erstellt, welcher die ausführende Person $o(a)$, sowie alle gefunden Ressourcen $\mathfrak{res}(a)$ enthält. Dieser Eintrag wird der Ressourcen-Menge $\mathfrak{Res}_a$ hinzugefügt.

Ist für eine Aktivität a bereits ein Ressourcen-Set $\mathfrak{res}_i^a \in \mathfrak{Res}_a$ bekannt, mit dem a arbeiten kann und es taucht ein weiterer Eintrag für a mit anderen Ressourcen auf, so wird ein neues Ressourcen-Set $\mathfrak{res}_j^a$ erzeugt. Aus den Log-Dateien lassen sich so die einzelnen Ressourcen-Konfigurationen $\mathfrak{Res}_a$ erlernen, mit denen eine Aktivität a ausführbar ist.

Algorithmus 4.4 : Ressourcennutzung aus Prozess-Logs extrahieren

Eingabe : Prozess-Log P
Ausgabe : Ressourcen-Menge $\mathfrak{Res}$

```
1 foreach a ∈ P do                                  // Für alle Aktivitäten
2     res ← o(a) ∪ res(a);          // Verwendete Ressourcen für Aktivität a
3     if res ∉ Res_a then
          // Ressourcen der Ressourcenmenge von a hinzufügen
4         Res_a.add(res);
5     end
6 end
7 return Res
```

Weitere mögliche Datenquellen

Die Ressourcennutzung einzelner Aktivitäten kann auch durch das Hinzuziehen weiterer Quellen erkannt werden. So wäre es zum Beispiel denkbar, dass Protokolle eines Servers über die genaue Auslastung der Hardware Aufschluss geben. Diese Information ließe sich mit denen aus Prozess-Logs kombinieren. Damit wäre es möglich zu berechnen, wie die Auslastung durch die gleichzeitige Bearbeitung mehrerer Aktivitäten beeinflusst wird. So lange die Auslastung unter 1 liegt, ist die entsprechende Ressource bereit, neue Anfragen zu bearbeiten und den Aktivitäten zur Verfügung zu stehen. Wird die Grenze der maximalen Auslastung jedoch erreicht, können weitere Aktivitäten, die auf diese Ressource angewiesen sind, nicht mehr bedient werden, bis die Ressourcennutzung unter den Grenzwert sinkt.

Eine Abschätzung dieses Grenzwertes befähigt die Simulation, genauere Ergebnisse unter Berücksichtigung der eingesetzten Ressource zu erzielen. Eine Abschätzung

der Kapazität einer Ressource durch Log-Dateien ist möglich, wenn angenommen wird, dass die gesamte Prozessarchitektur in den Prozess-Logs erfasst ist und die entsprechende Ressource ausschließlich von den erfassten Prozessen verwendet wird.

Beispiel

Im Beispiel aus Abbildung 4.11 sind insgesamt acht Ressourcen involviert. Diese sind die Ausführende Personen “Max Mustermann”, “Maria Musterfrau”, “Rudolf Ren” und “Erika Engel” sowie die Betriebs- und Hilfsmittel “Datenbank”, “Postfach”, “Banking” und “E-Mail”.

Die zeitliche Verwendung der Ressourcen aus diesem Beispiel wird in Abbildung 4.12 dargestellt.

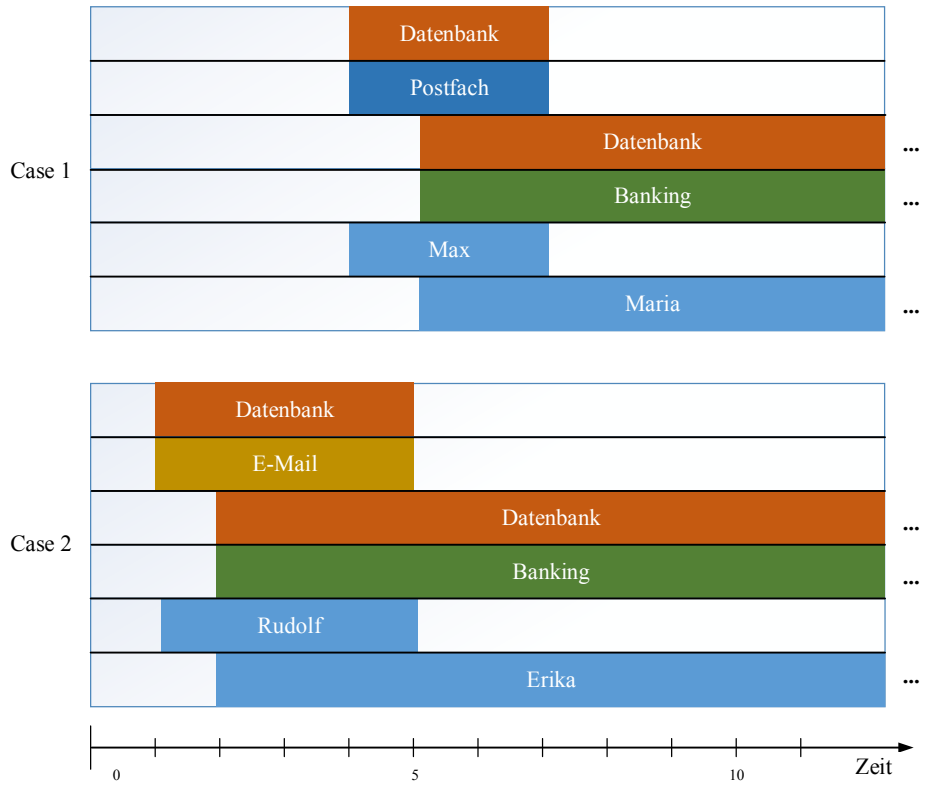


Abbildung 4.12.: Zeitliche Verwendung von Ressourcen durch zwei Instanzen eines Prozesses

Hier zeigt sich, dass diese Ressourcen innerhalb der beispielhaften Log-Datei unterschiedliche Verhalten aufweisen. In Case 1 wird die Datenbank im Zeitintervall 4-7

als auch ab Zeitpunkt 5 verwendet, wird also von mehreren Aktivitäten gleichzeitig genutzt.

Zusätzlich greift die zweite, zur gleichen Zeit ablaufende Instanz ebenfalls simultan auf die Datenbank zu. Es kommt zum Zeitpunkt 4 zu drei gleichzeitigen Zugriffen. Die minimale Kapazität der Datenbank in diesem Beispiel liegt also bei drei gleichzeitigen Zugriffen. Auch im zweiten Fall wird mehrfach auf die Datenbank zugegriffen. Die Ressource Datenbank stellt somit eine geteilte Ressource dar.

Im Gegensatz dazu wird das Postfach im Beispiel innerhalb des Prozesses lediglich exklusiv genutzt. Auch mit der zweiten Instanz findet keine zeitliche Überschneidung statt. Die gleichzeitige Nutzung einer Ressource kann aus einer anderen Instanz des selben Prozesses stammen (wie im Beispiel), oder auf Grund einer weiteren Log-Datei eines anderen Prozesses erkannt werden.

Angenommen, ein Protokoll der Serverauslastung der Datenbank wäre vorhanden, dann wäre es möglich, die maximale Auslastung der Datenbank zu ermitteln. Sei die in Abbildung 4.13 angegebene Kurve die aus Protokollen extrahierte Auslastung der Datenbank, ließe sich schließen, dass jeder Zugriff auf die Datenbank durch eine Aktivität die Serverauslastung um 10 % erhöht. Damit ließe sich die maximale Auslastung der Datenbank für die Simulation bestimmen. In diesem Beispiel würden zehn gleichzeitige Zugriffe die Datenbank vollständig auslasten.

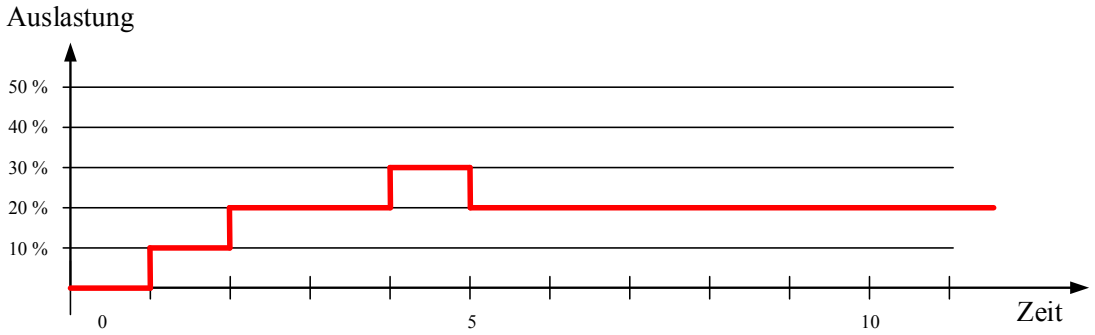


Abbildung 4.13.: Zeitliche Auslastung der Datenbank

4.8.4 Zeit-Mining

Prozess-Logs beinhalten Daten über die Abarbeitungszeit einzelner Prozesse. In diesem Absatz werden die Methoden dargestellt, um diese Daten für den Zeitkon-

text der RTPN-Modelle aufzubereiten und somit für die Simulation nutzbar zu machen.

Zeiten bestimmen

Ähnlich den verwendeten Ressourcen in einem Geschäftsprozess lässt sich das zeitliche Verhalten einzelner Aktivitäten nicht zur Gänze als Verteilungsfunktion beschreiben. Aktivitäten benötigen je nach ausführender Person oder Ressourcen unterschiedlich lange. Auch unter der Annahme gleicher Konfiguration unterliegt die Zeitdauer von Aktivitäten Schwankungen. Mit jeder Ausführung einer Aktivität werden neue Log-Daten erzeugt, die das tatsächliche Verhalten widerspiegeln. So zeigen gleiche Aktivitäten im Prozess-Log unterschiedliche, zeitliche Verhalten. Diese Schwankungen sind teilweise auf die verwendeten Ressourcen, dem ausführenden Subjekt oder auf andere, zufällige Faktoren zurückzuführen.

Das Verfahren für die Bestimmung des Zeitverhaltens aus Prozess-Logs entspricht der Inversions-Methode (Devroye 1986), welche es ermöglicht, die gefundenen Zeitwerte als Zufallsvariable abzubilden.

Zeitabbildung in Prozesslogs

Prozess-Logs besitzen die zeitlichen Informationen wie Start, Ende und Dauer $(t_s(a), t_e(a), t_d(a))$ für jede Aktivität $a \in A^*$ (siehe Kapitel 4.8.2). Aktivitäten in einem Prozess werden jedoch nicht ausschließlich gestartet und daraufhin erfolgreich beendet. Stattdessen können betriebliche Umstände dazu führen, dass Aktivitäten pausieren, die Arbeit nach einer gewissen Zeit wieder aufgenommen wird, oder Aktivitäten abbrechen, ohne dass sie ihr Ende erreichen. In einem Prozess-Log sind diese Zeitpunkte je einzeln aufgeführt, wie in Abbildung 4.14 beispielhaft für drei Aktivitäten abgebildet.

Diese unterschiedlichen Fälle müssen entsprechend behandelt werden, um daraus zeitliche Informationen zu ziehen. Aktivität 1 startet in dem Beispiel aus der Abbildung 4.14 in Zeitschritt 4 und endet in Zeitschritt 21, zeigt dazwischen jedoch eine Pausierung beginnend von Zeitschritt 7 bis Zeitschritt 10. Die Gesamtdauer von Aktivität 1 ließe sich als 17 Zeitschritte oder 14 Zeitschritte ermitteln, je nachdem, ob die Pausierung der Aktivität gezählt wird. Für Aktivität 2 lässt sich

indessen keine Zeit berechnen, da die Aktivität erfolglos abgebrochen wird. Wie lange die Aktivität noch bis zu Vollendung gebraucht hätte, lässt sich nicht bestimmen. Aktivität 3 zeigt ein Verhalten ohne Pausierung, mit einem Startzeitpunkt bei dem 12. Zeitschritt und dem Endzeitpunkt bei 17 Zeitschritten, weswegen die Zeitdauer der Aktivität 5 Zeiteinheiten beträgt.

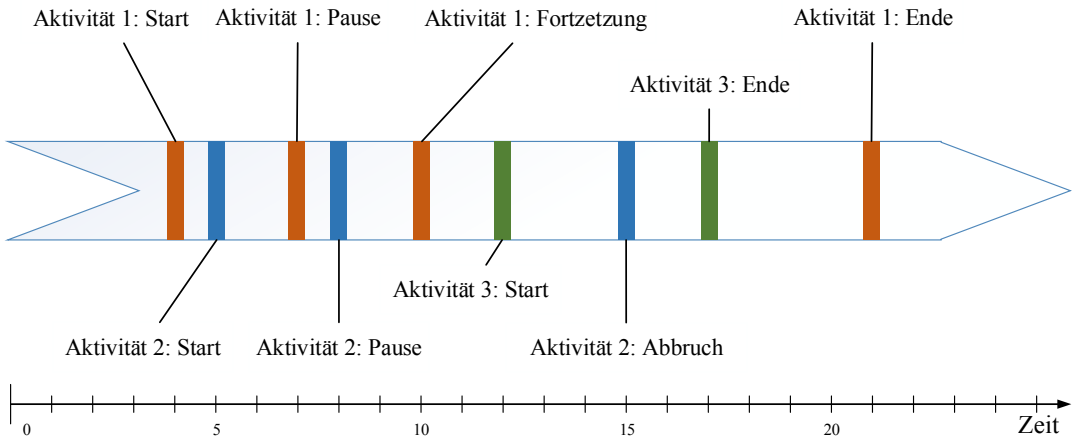


Abbildung 4.14.: Zeitliche Darstellung einer Log-Datei mit drei Aktivitäten und verschiedenen Ereignissen (Start, Pause, Fortsetzung, Abbruch und Ende).

Mindestvoraussetzungen

Wie im Beispiel beschrieben, müssen für eine Auswertung der Zeitinformationen aus Log-Dateien mindestens Start- und Endzeitpunkte der Aktivitäten im Log hinterlegt sein. Log-Dateien, in denen zum Beispiel nur der Zeitpunkt der Beendigung hinterlegt ist, lassen sich nicht verwenden, da die Informationen über die tatsächliche Dauer der Aktivität nicht vorhanden sind. Die Aktivitäten müssen dabei eindeutig identifizierbar sein, um eine Zuordnung zu ermöglichen (im Beispiel nummeriert von 1 bis 3). Dies gilt insbesondere für Aktivitäten mit gleichem Namen, da Aktivitäten mehrmals ausgeführt werden können oder die gleiche Aktivität in zusätzlichen, nebenläufigen Prozessen mehrfach gestartet werden kann.

Start einer Aktivität

Der Start einer Aktivität a muss entsprechend in den Log-Dateien markiert sein. Handelt es sich um nebenläufige Aktivitäten, so kann diese mehrmals gestartet

werden, noch bevor eine der bereits gestarteten Aktivitäten beendet wurde. Es ist in diesen Fällen notwendig, dass die einzelnen Instanzen in den Log-Dateien voneinander unterschieden werden können. Die Log-Dateien müssen Informationen über die tatsächliche Instanz beinhalten. In den in Kapitel 4.8.2 genannten Log-Formaten wird dies über eine eindeutige Instanz-ID gewährleistet. Der Startzeitpunkt wird als $t_s(a)$ gekennzeichnet.

Ende einer Aktivität

Das erfolgreiche Ende einer Aktivität a muss durch den Prozess-Log entsprechend gekennzeichnet sein. Im Falle mehrerer Instanzen einer einzelnen Aktivität muss die Instanz-ID vorhanden sein, um den Endzeitpunkt mit dem Startzeitpunkt dieser Instanz zusammenführen zu können. Der Endzeitpunkt der Aktivität a wird als $t_e(a)$ gekennzeichnet.

Pausierung und Wiederaufnahme einer Aktivität

Aktivitäten können pausieren. Unterschiedliche Gründe zwingen eine Aktivität zur Pausierung. Dies können eine Arbeitspause, Schichtwechsel, wechselnde Priorisierungen oder unerwartete Ereignisse wie Ausfälle oder dringliche, neue Aktivitäten sein. Die tatsächliche Zeit, an denen an der Aktivität jedoch bis zur Vollendung gearbeitet werden muss, ist dadurch nicht betroffen. Während den Pausen ruht die Aktivität.

Abgebrochene Aktivitäten

Abgebrochene Aktivitäten haben keine Dauer $t_d(a)$, da sie nie die Zustandsänderung “complete” durchlaufen (siehe Abbildung 4.10). Einträge ohne erfolgreichen Abschluss lassen sich nicht zur Extraktion des Zeitverhaltens der Aktivität nutzen, da sie kein normales Verhalten darstellen. Die Simulation durch RTPN-Modell stellt die Situation für korrekt durchgeführte Prozesse dar. Einträge von abgebrochenen Aktivitäten werden daher verworfen.

Häufig abbrechende Aktivitäten weißen auf ein erhöhtes Ausführungsrisiko der Prozesse dar. Dieses Risiko ließe sich in zukünftigen Arbeiten innerhalb der Simulation berücksichtigen.

Aktivitätsdauer

Die Zeitdauer $t_d(a)$ einer Aktivität ergibt sich damit aus der Differenz beider Zeitpunkte, wie bereits in 4.8.2 beschrieben:

$$t_d(a) = t_e(a) - t_s(a)$$

Im Falle von pausierten Aktivitäten gibt die Formel die Zeitdauer inklusive Pausen und anderen Unterbrechungen an. Alternativ könnten die unproduktiven Zeitintervalle von $t_d(a)$ abgezogen werden um die tatsächlich erbrachte Arbeitszeit ohne Pausierungen als Zeitdauer heranzuziehen. Allerdings spiegelt das Pausieren und Wiederaufnehmen der Arbeit das tatsächliche Zeitverhalten im Unternehmensumfeld wieder. Um die Simulation der Zeitdauer eines Prozesses möglichst realitätsnah zu gestalten, wird die oben genannte Formel ohne Änderung übernommen und genutzt, um den Zeitkontext der RTPN-Modelle zu füllen.

Erzeugen einer Zufallsvariable mit dem Simulationslemma

Wie beschrieben, lassen sich aus den Prozess-Logs Stichproben des Zeitverhaltens einzelner Aktivitäten bestimmen und zur Visualisierung als Histogramm aufzeichnen. Für die Simulation des Zeitverhaltens sind Zufallsziehungen aus den beobachteten Stichproben notwendig. Ein entsprechend der Beobachtung agierender Zufallsgenerator lässt sich durch die sogenannte Inversionsmethode oder Quantil-Transformation erstellen (Devroye 1986; Georgii 2015). Dabei werden die einzelnen Werte des Histogramms normalisiert und integriert. Die Inversionsmethode ist ein Verfahren, um aus gleichverteilten Zufallszahlen andere Wahrscheinlichkeitsverteilungen durch die Inverse der Zufallsfunktion zu generieren und basiert auf den Simulationslemma 4.1 und 4.2:

Lemma 4.1. *Sei X eine stetige Zufallsvariable mit der Verteilungsfunktion F_X, dann zeigt die Zufallsvariable $Y = F_X(X)$ eine Gleichverteilung im Intervall $[0, 1]$.* ⊣

Die Inversionsmethode ist die Umkehrung dieser Definition:

Lemma 4.2. *Ist Y eine gegebene gleichverteile Zufallsvariable im Intervall $[0, 1]$ und X hat die Verteilungsfunktion F_X, dann hat die Zufallsvariable $F_X^{-1}(Y)$ die gleiche Verteilung wie X.* ⊣

Beweis: Die erste Behauptung folgt für $x \in \mathbb{R}$,

$$P(F^{-1}(U) \leq x) = P(inf\{y|F(y) = U\} \leq x) = P(U \leq F(x)) = F(x)$$

Die zweite Behauptung folgt aus der Tatsache, dass für alle $0 < u < 1$ gilt,

$$P(F(X) \leq u) = P(X \leq F^{-1}(u)) = F(F^{-1}(u)) = u$$

Die Inversionsmethode ist nach (Devroye 1986) eine universelle und somit stets anwendbare Methode, eine beliebige Zufallsvariable zu erzeugen. Verwandte Verfahren benötigen zusätzliche Informationen über die zu Grunde liegende Population. Da die Verteilung der Zeitdauern von Aktivitäten nicht zwangsläufig einer bestehenden, stochastischen Verteilung folgen müssen, ist die Inversionsmethode somit das gewählte Verfahren, eine Zufallsvariable mit dem entsprechendem Verhalten aus Log-Dateien zu generieren.

Die Verwerfungsmethode (Georgii 2015) ist ein weiteres, einsetzbares Verfahren zur Erstellung einer Zufallsvariable mit gegebener Wahrscheinlichkeitsdichte. Sie zeigt jedoch ein schlechteres Zeitverhalten, da diese Verwerfungsmethode algorithmisch auf dem mehrmalige Durchlaufen einer Schleife beruht.

Die Inversionsmethode lässt sich somit nutzen, um aus gegebenen Log-Dateien eine Zufallsvariable mit der zu Grunde liegenden Verteilung zu approximieren.

Verfahren

Die Inversionsmethode kann genutzt werden, solange die Umkehrfunktion F_X^{-1} berechenbar ist. Für diskrete Verteilungen kann zwar nicht die Inverse gebildet werden, aber eine leichte Modifikation der Inversionsmethode lässt sich auch zur Erzeugung diskret verteilter Pseudozufallszahlen anwenden, siehe Algorithmus 4.5. Aus den einzelnen Zeiteinträgen der Log-Dateien lässt sich die diskrete Wahrscheinlichkeitsdichte $f(X)$ (Zeile 8 aus Algorithmus 4.5) und somit die umkehrbare, diskrete Verteilungsfunktion $F(X)$ extrahieren (Zeilen 15 und 18 aus Algorithmus 4.5).

Algorithmus 4.5 : Verfahren zum Erzeugen der diskreten Inverse aus einem gegebenen Stichprobensatz

1 Inversionsmethode (T, n);
  Eingabe : T: Array mit Stichproben
  h: Anzahl gewünschter Balken (Histogrammklassen) (Gl. 4.2)
  Ausgabe : F^{-1}: Inverse Wahrscheinlichkeitsverteilung passend zu T
2 min,max ← Minimum und Maximum von T;
3 size ← (max − min)/h // Größe der Klassen
4 f ← initialisiere 0-Array mit h +1 Einträgen ; // $f(X)$
5 F ← initialisiere Array mit h +1 Einträgen ; // $F(X)$
6 $F^{-1} \leftarrow \emptyset$; // F_X^{-1}
7 **foreach** $t \in T$ **do** // Zähle Einträge in T
8 f$[\lfloor \frac{t-\text{min}}{\text{size}} \rfloor]$++ ; // Inkrementiere zu t passende Histogrammklasse
9 **end**
10 F[0]← f[0]
11 **for** $i \leftarrow 1$ **to** h + 1 **do** // Summiere Einträge aus f
12 F[i]←F$[i-1]$+f[i] ;
13 **end**
14 **for** $i \leftarrow 0$ **to** h + 1 **do** // Normiere Einträge in F
15 F[i]← $\frac{F[i]}{F[\text{h}+1]}$;
16 **end**
17 **for** $i \leftarrow 0$ **to** h + 1 **do** // Erzeuge Punktmenge
18 $F^{-1} \leftarrow F^{-1} \cup (F[i], \text{f}[i])$; // Füge x/y Koordinate zu F^{-1} hinzu
19 **end**
20 **return** F^{-1}

Das Verfahren lässt sich für jede Aktivität, die in den Prozess-Logs vorkommt, durchführen. Dabei muss die Aktivität mindestens einmal erfolgreich durchgeführt worden sein, damit die Aktivitätsdauer gemäß Abschnitt 4.8.2 berechenbar ist.

Je nach Wahl der Balkenbreite lässt sich zwischen erhöhtem Speicherbedarf und Genauigkeit oder höherer Generalisierung und verringertem Speicherbedarf wählen. Eine geringe Anzahl an Histogramm-Klassen führt dazu, dass mehrere Werte zusammengefasst werden. Der Speicherbedarf der Variablen f, F und letztendlich der Ausgabe F^{-1} (Zeilen 4-6 im gezeigten Algorithmus 4.5) sinkt entsprechend. Dabei verringert sich die Anzahl der möglichen Ausgabewerte der Inversionsmethode. Die Inversionsmethode kann somit Datenmengen variablen Umfangs zusammenfassen. Dadurch lassen sich aus Prozess-Logs unterschiedlicher Größe mit Hilfe des Simulationslemma ein Zufallsgenerator erstellen, der für die Simulation nutzbar ist.

Eine geeignete Wahl der Menge der Histogramm-Klassen und somit der Generalisierung der Daten ist abhängig von der Menge der Einträge der Log-Dateien und lässt sich nach der Regel von Freedman und Diaconis ableiten (Freedman et al. 1981). Diese Regel berechnet die Anzahl h der Klassen (oder Balken) als:

$$h = \frac{2 \cdot (Q_3 - Q_1)}{\sqrt[3]{n}} \tag{4.2}$$

wobei Q_1 und Q_3 das erste und dritte Quartil der Stichproben aus den Log-Files darstellen. Die Quartile Q_x mit $x = \{1, 2, 3\}$ beschreiben den Punkt einer Verteilung oder Datenreihe, unterhalb dessen $\frac{1}{4}, \frac{2}{4}$ oder $\frac{3}{4}$ aller Punkte der Datenreihe vorhanden sind (Georgii 2015).

Beispiel

Angenommen, ein Prozess-Log liefert für eine Aktivität die zehn angegebenen Zeitdauern:

5 2 2 5 7 2 9 5 9 5

Die reinen Häufigkeiten der einzelnen Zeiteinträge sind in Abbildung 4.15 als Histogramm abgebildet. Dieser Schritt entspricht den Zeilen 7-9 aus Algorithmus 4.5. Als Anzahl der Klassen wurde 8 gewählt (Klasse 2 - 9), womit sich eine Klassengröße von 1 ergibt.

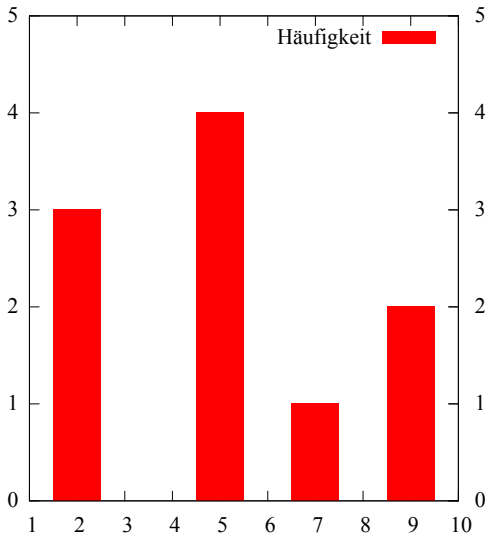


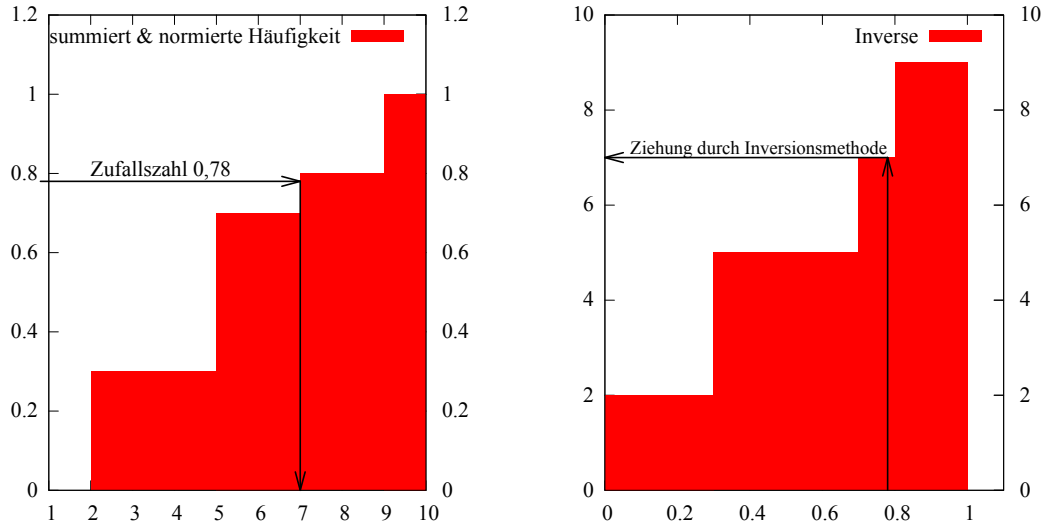
Abbildung 4.15.: Histogramm der Beispiel-Datenreihe mit Klassengröße 1.

Das Verfahren zur Erzeugung der Inversen läuft entsprechend Algorithmus 4.5 wie folgt ab:

- Zeilen 1-6: Initialisierung der Variablen und Berechnung der Klassengröße.
- Zeilen 7-9: Die Erzeugung des Histogramms, wie in Abbildung 4.15 dargestellt.
- Zeilen 11-16: Die Häufigkeiten werden summiert und normiert, um somit die kumulierte Verteilungsfunktion F_X zu erhalten. Das summierte und normierte Histogramm in Abbildung 4.16a zeigt F_X.
- Zeilen 17-19: Die Inverse F_X^{-1} entsteht, indem die Achsen vertauscht werden. Die entstehende Inverse ist im Beispiel in Abbildung 4.16b zu sehen.
- Zeile 20: Aus der erzeugten Inverse F^{-1} können nun Zufallszahlen gezogen werden.

In diesem Beispiel wird von einem gleichverteilten Zufallsgenerator (entsprechend der Zufallsvariable Y aus Lemma 4.2) der Wert $0,78$ gewählt. Das Ergebnis der Inversionsmethode gibt daraufhin den Wert 7 für F_X^{-1}zurück, entsprechend dem inversen Ergebnis $F_X^{-1}(0,78) = 7$. In dem Beispiel zeigt sich, dass die Zahl 7 insgesamt nur ein Mal vorkommt. Auf der Ordinate "belegt" daher Zahl 7 nur einen kleinen Abschnitt, hat also entsprechend ihrer Häufigkeit in dem Datensatz eine

geringe Wahrscheinlichkeit von dem gleichverteilten Zufallsgenerator auf dem Intervall [0...1] gezogen zu werden. Die Werte 2 und 5 hingegen kommen häufig vor und haben entsprechend eine höhere Wahrscheinlichkeit, von einem gleichverteilten Zufallsgenerator "gezogen" zu werden. Die "Ziehung" der Inverse durch einen gleichverteilten Zufallsgenerator ist sowohl in der kumulierten Verteilungsfunktion in Abb. 4.16a als auch der tatsächlichen Inversen in Abb. 4.16b mit Pfeilen dargestellt.



(a) Kumuliertes Histogramm der Beispiel-Datenreihe

(b) Ergebnis der Inversionsmethode der Beispiel-Datenreihe

Abbildung 4.16.: Funktionsweise der Inversionsmethode für das wählen einer Zufallszahl aus einer diskreten Verteilung

Das Beispiel zeigt die Möglichkeit, das Zeitverhalten der Aktivitäten aus den Prozess-Logs in den Zeit-Kontext der RTP-Nets zu überführen. In Kombination mit dem Verfahren zur Extraktion der Ressourcen-Nutzung aus Prozess-Logs lassen sich somit die Abhängigkeiten zwischen den Prozessen aufdecken, um die Auswirkungen durch Änderungen der Zugangskontrolle oder Prozessstruktur eines Prozesses auf andere Prozesse zu simulieren.

4.9 Zusammenfassung

In diesem Kapitel wurden die im Rahmen dieser Dissertation erarbeiteten Resource-Timed Petri Nets vorgestellt. Dies stellt ein Modell zur Abbildung von

Geschäftsprozessen mit dessen Zeit- und Ressourcenverhalten, als auch deren Sicherheitsanforderungen durch eine rollenbasierte Zugriffskontrolle dar. Zusammengefasst kombinieren und erweitern RTPN-Modelle die Funktionalität verschiedener Petri-Netz-basierten Modelle:

- RTPNs nutzen einen Zeitkontext, um beliebige Verteilungen als Abarbeitungszeit von Aktivitäten zu wählen. Dies stellt eine Verbesserungen der Modellierungsmöglichkeit gegenüber den Timed oder Stochastic Petri Nets dar, da diese lediglich parametrisierte Intervalle oder Exponentialfunktionen als Zeitfunktion nutzten.

- Die Repräsentation von Ressourcen erfolgt in RTPNs über einen Ressourcenkontext. Die Darstellung geteilter und exklusiv zugreifbarer Ressourcen wird hiermit ermöglicht. Die Fähigkeit dieses Kontextes erlaubt im Gegensatz zu CPNs die Zuweisung unterschiedlicher Ressourcen-Mengen, die zur Abarbeitung von Aktivitäten führen können.

- RTPN-Modelle nutzen den Zugriffskontext der IFNets für die Zuweisung der Ausführungsrechte einzelner Aktivitäten. Dieser Kontext modelliert die Zugriffsrechte durch ein Rollenkonzept, um Interessenkonflikte durch Funktionstrennung oder Funktionsbindung zu behandeln.

Um die zeitliche Ressourcenabhängigkeit multipler Prozesse zu fassen, nutzen RTPN-Modelle eine global gültige Markierungsmenge. Hierdurch lassen sich im Gegensatz zu bestehenden Petri-Netz-basierten Simulationsmethoden ganze Prozessarchitekturen simulieren. Da RTPNs ein Rechtemodell abbilden, lassen sich auch die Auswirkungen von Änderungen der Sicherheits- und Compliancebestimmungen simulieren, wie zum Beispiel ein Vier-Augen-Prinzip oder die Funktionstrennung zur Vermeidung von Interessenkonflikten. Die Auswirkungen geänderter Zugriffskontrollen, sowie geänderter Kontrollflüsse oder Ressourcennutzungen lassen sich somit für eine gesamte Prozessarchitektur statt nur für einzelne Prozesse simulieren.

Die Ergebnisse aus der Simulation eines Resource-Timed Petri Nets sind umso aussagekräftiger, desto exakter die Ausgestaltung des Zeit- und Ressourcenkontextes ist. Zu diesem Zweck wurden im Rahmen dieser Dissertation Verfahren entwickelt, die Kontexte der RTPNs aus Prozess-Logs zu füllen, um reale Daten der Prozessabläufe nutzbar zu machen. Die hieraus resultierenden, realistischen

Annahmen über das Ressourcen- und Zeitverhalten der Prozesse ermöglichen eine zutreffende Simulation der Prozesse durch RTPN-Modelle.

Auswirkungen von Prozessänderungen sind damit noch vor dem produktiven Einsatz ermittelbar. Die RTP-Nets und die vorgestellten Verfahren zur Kontextgenerierung aus Prozess-Logs beantworten somit die Forschungsfragen, da unterschiedliche Umsetzungsmöglichkeiten notwendiger Prozessänderungen an Hand realistischer Daten evaluierbar werden. Dies versetzt Prozessdesigner und Entwickler in die Lage, Änderungen so umzusetzen, dass die in einer Prozessarchitektur verursachten Kosten und Auswirkungen auf die Verfügbarkeit der Prozesse minimal sind. Damit entschärft sich das Spannungsfeld zwischen Sicherheits- oder Compliancemaßnahmen und der Verfügbarkeit und fördert Akzeptanz der Stakeholder zugunsten der Umsetzung sicherheitsrelevanter Prozessanpassungen.

Kapitel 5

Evaluation

Nachdem die entwickelten RTPN-Modelle, das zugehörige Simulationsverfahren und die Generierung des Zeit- und Ressourcenkontexts gezeigt wurden, wird in diesem Kapitel die Evaluation der RTPN-Modelle anhand der Simulation von Prozessänderungen, die der Sicherheit dienlich sind, gezeigt. Abschließend wird der Einsatz von RTPNs zur optimierten Steuerung dieser, nun sicher gestalteten Geschäftsprozessen präsentiert. Hierfür wurden die RTPN-Definition sowie Simulations- und Optimierungsfunktionalitäten innerhalb der Softwareumgebung des Security Workflow Analysis Toolkit (SWAT) implementiert. SWAT ist eine Plattform für die Modellierung und Analyse von Geschäftsprozessen in Form von Petri-Netzen und wurde am Institut für Informatik und Gesellschaft der Albert-Ludwigs-Universität Freiburg entwickelt (Zahoransky et al. 2016; Accorsi et al. 2014). Die gezeigten Beispiele, an Hand deren die Evaluation durchgeführt wird, stellen eine Kombination von realen Daten deutscher Unternehmen sowie von künstlich erzeugten Prozessen dar. Für die Generierung der Ressourcen- und Zeitkontexte aus den Prozess-Logs wurden im Falle existierender Log-Dateien die in Kapitel 4.8 gezeigten Verfahren genutzt.

Die Ergebnisse der Evaluation sollen die Einsatzfähigkeit und Möglichkeiten der Simulation von Geschäftsprozessen durch Resource-Timed Petri Nets zeigen. Hierfür wird zuerst die gegenseitige Beeinflussung nebenläufiger Prozesse dargestellt. Daraufhin wird mit Hilfe des Simulationsverfahrens gezeigt, dass sich Änderungen eines Prozesses negativ auf die Abarbeitung eines anderen Prozesses auswirken können und Sicherheits- und Compliancebestimmungen (wie die Aufgabentrennung) diese negativen Auswirkungen weiter verstärken. Um diese Störungen

verständlich zu quantifizieren, werden diese Auswirkungen sowohl als prozentuale Angaben als auch als monetärer Wert angegeben, um die erhöhten Kosten durch Sicherheitsmaßnahmen zu zeigen.

Als Ausblick folgt die Demonstration des Optimierungsverfahren, welches die Abarbeitungsreihenfolge der Aktivitäten so anordnet, dass Störungen zwischen den Prozessen minimiert werden. Hierdurch lässt sich eine Prozessarchitektur steuern, damit sie möglichst ohne Ressourcenkonflikte arbeitet. Dies führt zu geringeren Kosten der Sicherheits- oder Complianceimplementierung. Diese Tatsache ermöglicht es zum einen, Compliance oder Sicherheitsbestimmungen mit möglichst geringen Kosten umzusetzen und Sicherheitsmaßnahmen durchzuführen, welche das Unternehmen sonst zu stark belasten würden.

5.1 Secure Workflow Analysis Toolkit

Für die Evaluation der RTPN-Modelle wurde die Software-Plattform SWAT gewählt. SWAT wurde am Institut für Informatik und Gesellschaft im Rahmen des Projekts "Geschäftsprozess-Sicherheit zur Verstärkung des Einsatzes von e-Business-Standards" (GESINE) (Eymann et al. 2014) entwickelt und bietet Funktionalitäten zur Sicherheits- und Verfügbarkeitsanalyse von Geschätsprozessen und Prozess-Logs (Zahoransky et al. 2016; Accorsi et al. 2014). SWAT wurde in Java in der Version 7 implementiert und unter Windows, MacOS und Linux Betriebsystemen getestet und entwickelt. Veröffentlicht ist SWAT auf der Entwicklerplattform GitHub[1]. Als modular entwickelte Software bietet SWAT seine Funktionen und Schnittstellen als Pakete an. Damit stellt SWAT eine Forschungs- und Entwicklungsplattform für weitere und neue Analyse- und Simulationsmethoden von Geschäftsprozessen oder Prozess-Logs dar.

Hierfür kapselt das dedizierte Paket "SEWOL" der SWAT das Einlesen und Verarbeiten von Prozess-Logs in unterschiedlichen Formaten wie XES oder MXML. Geschäftsprozesse kann SWAT als BPMN importieren und teilautomatisiert zu einem Petri-Netz konvertieren. Die Darstellung von Geschäftsprozessen erfolgt als Petri-Netz durch das Security-oriented Petri Net Framework (SEPIA), einem weiteren Paket innerhalb von SWAT. Diese Pakete sind ebenfalls Eigenentwicklungen

[1]SWAT 2.0, Version 0.0.2: https://github.com/iig-uni-freiburg

des Instituts für Informatik und Gesellschaft. Neben einfachen Petri-Netzen kann SEPIA Colored Petri-Nets und Information-Flow Petri-Nets einlesen und modellieren. Als Editor für Petri-Netze nutzt SWAT “Wolfgang”, einen im Rahmen von SWAT entwickelten Editor mit Funktionen zur Darstellung und Manipulation von Petri-Netzen in unterschiedlichen Ausprägungen. Die Darstellung von Prozess-Logs erfolgt textuell. SWAT bietet somit Analyse-Methoden sowohl für Prozess-Logs als auch für Petri-Netze.

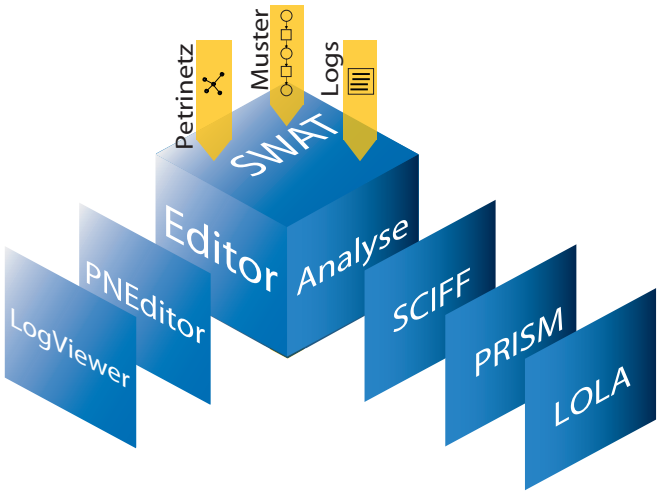


Abbildung 5.1.: Übersicht über die Module von SWAT

SWAT wurde bereits erfolgreich zur Evaluation neuer Analysemethoden benutzt. So wurden die IFNets samt derer Analyseverfahren erstmals in SWAT implementiert. Weitere bekannte Verfahren wurden erfolgreich in SWAT adaptiert, so zum Beispiel das abduktive Prüfverfahren SCIFF der Universität Bozen (Alberti et al. 2008) zur Sicherheit- und Complianceanalyse von Prozess-Logs. Das Ziel von SWAT ist es, die genannten Prüfverfahren für End-Nutzer zu Verfügung zu stellen und vorgefertigte Analysemethoden zu integrieren, damit die bestehenden Verfahren einer breiten Masse an Benutzern verfügbar gemacht werden kann. Diese einzelnen fremd- und selbstentwickelten, auch unabhängig voneinander verwendbare Module sind in Grafik 5.1 abgebildet und als SWAT in einer grafischen Oberfläche vereint. Abbildung 5.2 zeigt die Benutzeroberfläche von SWAT.

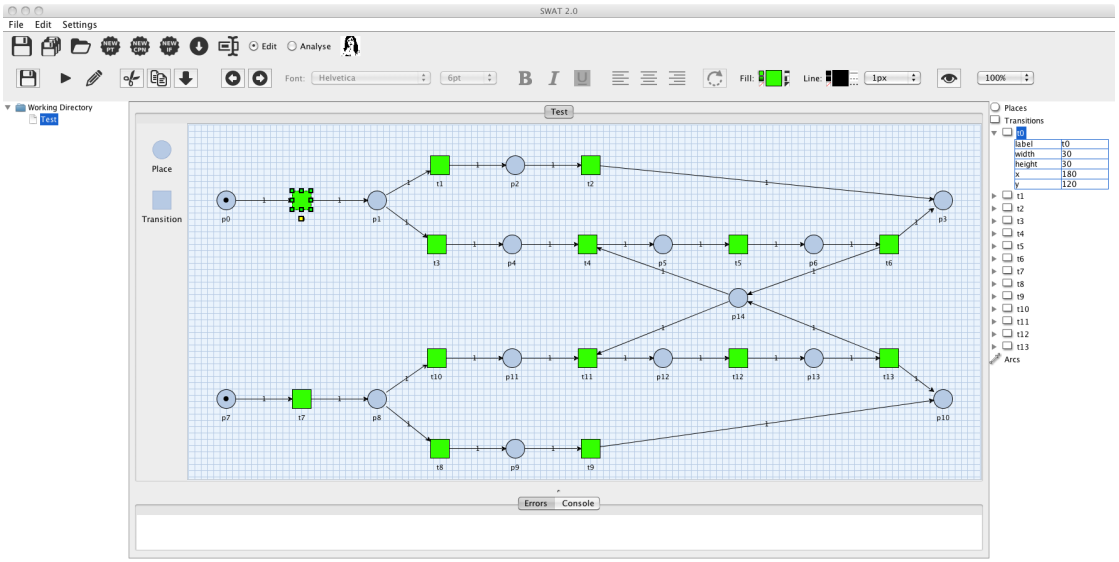


Abbildung 5.2.: Screenshot von SWAT

5.1.1 Security-oriented Petri Net Framework

Das Security-oriented Petri Net Framework (SEPIA) bietet Funktionalitäten für die Erzeugung, Bearbeitung, das Speichern sowie Laden und die Ausführung von Petri-Netzen. Die Ausführung eines Petri-Netzes stellt dabei das Token Game dar. Das Paket ermöglicht die Erstellung von Petri-Netzen, CPNs, IFNets und, durch diese Arbeit, von RTPNs.

Für die Evaluation wurden Netzdefinitionen für die RTPNs implementiert. Die RTP Nets wurden aufbauend auf der Petri-Netzdefinition aus SEPIA implementiert und nutzen den Zugriffskontext der IFNets. Die Feuerregeln der Petri-Netze wurden um die der RTPNs erweitert. Dafür wurden die Feuerbedingungen der Transitionen so verändert, dass der externe Ressourcen- und Zeitkontext beachtet wird. Eine Transition kann nun nur feuern, wenn die benötigten Ressourcen vorhanden sind. Bei jedem Feuern wird der Zeitkontext nach der Feuerdauer befragt. Statt dass die Transition die Tokens direkt erzeugt, wird die Transition in die Markierungsmenge M hinzugefügt. Es wurden Schnittstellen geschaffen, um den Ressourcen- und Zeitkontext zu modellieren.

Eine zusätzlich implementierte Klasse (`TimeMachine`) koordiniert das Simulationsverfahren der RTP Nets mit der zugehörigen Markierungsmenge und Kontexten. Diese Klasse setzt die Algorithmen 4.1 und 4.2 um.

5.1.2 Security-oriented Workflow Library

Neben der Analysen von Petri-Netzen, welche Geschäftsprozesse beschreiben, bietet SWAT die Möglichkeit zum Einlesen und Prüfen von Log-Dateien. Diese Funktionalität ist in dem Security-oriented Workflow Library (SEWOL) Paket implementiert. Das Paket unterstützt das Einlesen der bekannten Log-Formate und bietet Schnittstellen, diese Prozess-Logs zu iterieren und die Aktivitäten auszulesen. Dabei ist es möglich, mit SEWOL den Benutzer einer Aktivität ($o(a)$) die Zeitstempel ($t_s(a), t_e(a), t_d(a)$) sowie die verwendeten Ressourcen ($\mathfrak{res}(a)$) auszulesen.

SEWOL wurde um das Verfahren zur Ermittlung der Ressourcennutzung, bestehend aus den Algorithmen 4.3 und 4.4, als auch um das Inversionsverfahren aus Algorithmus 4.5 zur Erzeugung von Zufallszahlen aus Prozess-Logs erweitert. Hiermit ist es möglich, den Zeit- und Ressourcenkontext für die RTPN-Modelle automatisiert zu erstellen.

5.1.3 Ressourcen-Kontext

Der Ressourcen-Kontext wurde im Rahmen dieser Arbeit innerhalb von SWAT realisiert. Der Ressourcen-Kontext implementiert dabei die in SEPIA definierte Schnittstelle. Die Schnittstelle besteht aus folgenden Methoden:

```
public interface IResourceContext {
        public String getName();
        public void blockResources(List<String> resources);
        public void unBlockResources(List<String> resources);
        public List<String> getRandomResourceFor(String activity);
        public boolean isAvailable(String resourceName);
        public boolean isAvailable(List<String> resourceName);
        public IResource getResourceObject(String resourceName);
        public void reset();
        ...
}
```

Programmtext 5.1: Interface des Ressourcen-Kontext

Ressourcen-Kontexte besitzen einen Namen, da sie serialisierbar sind und sich speichern und laden lassen. Die Methoden der Schnittstelle müssen folgende Implementierung aufweisen:

- blockResources: Setzt die angegebenen Ressourcen als verwendet für exklusive Ressourcen oder erhöht die Auslastung für geteilte Ressourcen.
- unBlockResources: Setzt die angegebenen Ressourcen wieder frei oder verringert deren Auslastung.
- getRandomResourcesFor: Sucht einen zufälligen Satz an Ressourcen, mit welchen die angegebene Aktivität arbeiten kann. Die zurückgegebenen Ressourcen sind verfügbar.
- isAvailable: Gibt zurück, ob die angegebenen Ressourcen verfügbar sind und setzt hiermit die Funktionen $avail(\mathfrak{r})$ sowie $avail(\mathfrak{res})$ um.
- getResourceObject: Gibt die gesuchte Ressource als Ressourcen-Objekt zurück.
- reset: Setzt alle Ressourcen auf verfügbar.

Die Ressourcen werden als Ressourcen-Objekte `IResource` im Ressourcenkontext vorgehalten. Diese entsprechen dem Ressourcen-Typ aus Definition 4.5. Die Methode `getResourceObject` gibt ein Objekt dieses Typs zurück:

```
public interface IResource {
        public String getName();
        public boolean isAvailable();
        public void use();
        public void unUse();
        public void reset();
}
```

Programmtext 5.2: Interface der Ressourcenobjekte

Die Implementierung der Schnittstelle zeigt folgende Funktionalität:

- isAvailable: Ist wahr, wenn die Ressource zur Verfügung steht und entspricht damit der Funktion $avail(\mathfrak{r})$.

- use: Blockiert exklusiv genutzte Ressourcen oder erhöht die Auslastung der Ressource, falls sie vom Typ geteilt ist.
- unUse: Gibt die Ressource frei oder verringert die Auslastung.
- reset: Setzt die Ressource frei und reduziert die Auslastung auf 0.

Diese Schnittstellen bilden den Ressourcenkontext $\mathfrak{R}$ mit den einzelnen Einträgen $\mathfrak{Res} = \{\mathfrak{res}_i\}$ und den einzelnen Ressourcen $\mathfrak{r} \in \mathfrak{res}_i$ der RTP Nets um. Die Methoden `isAvailable` stellen die Funktion $avail(\mathfrak{r})$ und $avail(\mathfrak{res})$ dar.

5.1.4 Zeit-Kontext

Der Zeitkontext wurde im Rahmen dieser Arbeit ebenfalls innerhalb von SWAT realisiert. Analog zu dem Ressourcenkontext, implementiert der entwickelte Zeit-Kontext die Schnittstelle aus dem Paket SEPIA, welche folgende Methoden verlangt:

```
public interface ITimeContext {
        public double getTimeFor(String activity, List<String>
            resources);
        public ITimeBehaviour getTimeObjectFor(String activity);
        public boolean containsActivity(String activity);
}
```

Programmtext 5.3: Interface des Zeitkontexts.

Diese Methoden sind durch den Zeitkontext in SWAT mit folgenden Funktionalitäten implementiert:

- getTimeFor: Gibt für die angegebene Aktivität a und den Ressourcen $\mathfrak{res}$ eine zufällig gewählte Zeitdauer aus der hinterlegten Funktion $P(a, \mathfrak{res})$ zurück.
- getTimeObjectFor: Gibt die für die Aktivität a und Ressourcen $\mathfrak{res}$ die hinterlegte Zufallsvariable $\chi(a, \mathfrak{res})$ aus.
- containsActivity: Gibt an, ob für die Aktivität ein Eintrag existiert. Transitionen, die keinen Zeiteintrag besitzen, feuern gemäß den Feuerregeln der Petri-Netze ohne Wartezeiten.

Die dem Zeitkontext hinterlegten Objekte entsprechen der Schnittstelle `ITimeBehaviour`. Diese Objekte entsprechen den Einträgen $\Omega_{T_{\mathfrak{Res}}}$. Sie besitzen die Methode `getNeededTime`, welche einen Wert der einzelnen Einträge $d\ in \Omega_{T_{\mathfrak{Res}}}$ aus der Inversionsmethode oder einen Wert der stetigen Verteilung $pdf(x)$ zieht.

Hiermit ist die notwendige Funktionalität des Zeitkontextes gewährleistet.

5.1.5 Zugriffskontext

Der Zugriffskontext für die RTPNs wurde aus den IFNets des SEPIA-Frameworks übernommen und unterstützt die Verwendung sowohl von Zugriffskontrollmatrizen als auch einer erweiterten Rechteverwaltung durch ein RBAC Modell (Stocker et al. 2013).

5.2 Softwarearchitektur

Die Implementierung der RTPNs, des Zeit- und Ressourcenkontexts sowie des Simulationsverfahrens folgt dem Klassendiagramm aus Abbildung 5.3.

Der Ressourcenkontext `ResourceContext` implementiert die Schnittstelle `IResourceContext` und besteht aus einzelnen Objekten des Typs `IResource`. Der Zeitkontext `TimeContext`, definiert durch die Schnittstelle `ITimeContext`, umfasst `ITimeBehaviour`-Objekte. Ein einzelnes RTPN-Modell besteht aus jeweils einem dieser Kontexte. Der aus den IFNets entnommene Zugriffskontext ist ebenfalls Teil der RTPNs, jedoch nicht explizit aufgeführt.

Die Struktur eines RTPNs besteht aus den Transitionen `RTPNTransition`, den Plätzen `RTPNPlace` sowie den Flussrelationen `RTPNFlowRelation`. Die Klassen `RTPN`, `RTPNTransition`, `RTPNPlace` und `RTPNFlowRelation` erben ihre Funktionalität von den bereits existierenden Petri-Netz Modellen aus dem SEPIA-Paket und wurden um die Funktionalität und Feuerregeln der RTPN-Modelle erweitert.

Die Simulationsverfahren aus Algorithmus 4.1 und Algorithmus 4.2 zur Simulation der RTPNs sind durch die Klasse `TimeMachine` implementiert. Diese Klasse besitzt mindestens ein RTPN-Modell und umfasst mit `PendingActions` die Markierungsmenge M.

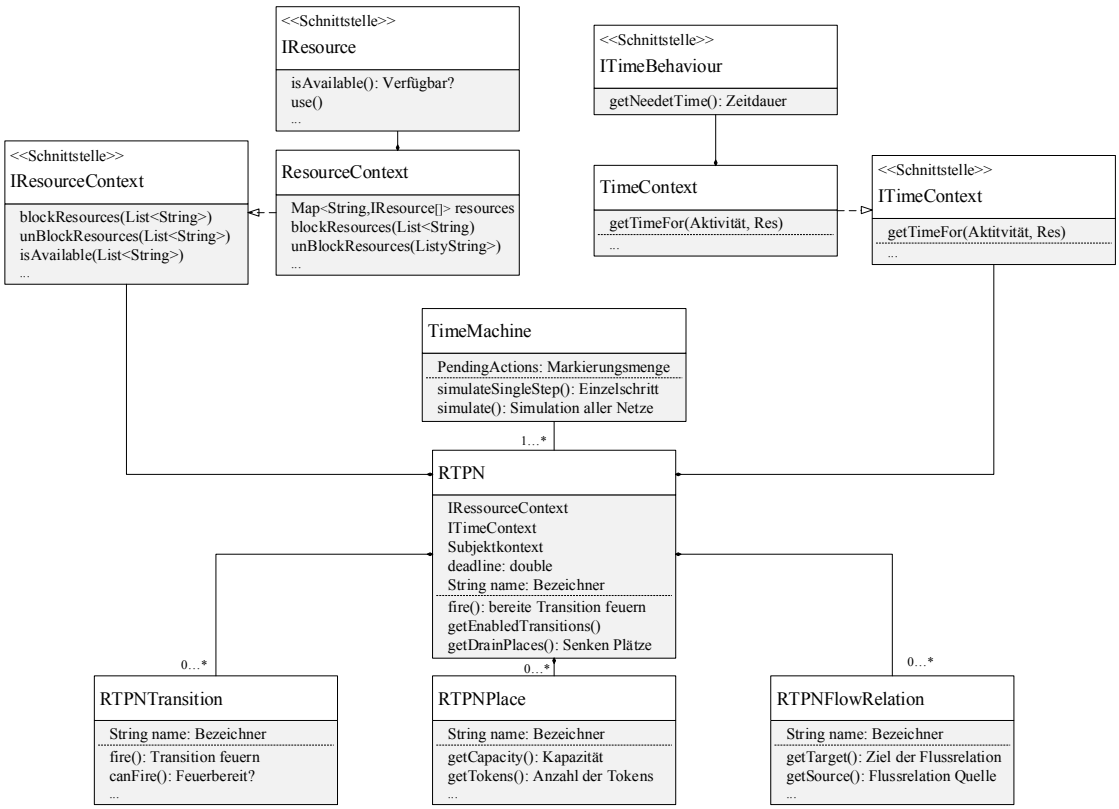


Abbildung 5.3.: Klassendiagramm der RTP Nets und TimeMachine.

5.3 Evaluation anhand einer Prozessarchitektur

Für die Evaluation durch die gezeigte Implementierung wurde eine der Praxis entsprechende Prozessarchitektur mit mehreren Prozessen entworfen. Diese Prozesse sind im Anhang B genauer beschrieben. Für die Evaluation wurden die Prozesse in RTPN-Modelle überführt. Somit lässt sich die Funktionalität der entwickelten Resource-Timed Petri Nets demonstrieren. Die im Folgenden vorgestellten Ergebnisse zeigen den Einfluss von Änderungen eines Prozesses auf andere Prozesse. Die in der Prozessarchitektur entstandenen Störungen lassen sich sowohl zeitlich, als auch prozentual darstellen. Für eine monetäre Darstellung wird das TD ABC, eine Möglichkeit zur zeitabhängigen Prozesskostenschätzung, auf die Simulationsergebnisse angewandt (Baltzer 2007; Reckenfelderbäumer 2013).

Die Simulation erzeugt pro Durchlauf eine zufällige Ablaufreihenfolge der Aktivitäten innerhalb der Prozessarchitektur. Die Ergebnisse aus wiederholten Simulationsabläufen lassen eine Optimierung auf Basis der entsprechenden Ergebnisse zu. Die Ergebnisse zeigen, dass unterschiedliche Abarbeitungsreihenfolgen zu un-

terschiedlichen, zeitlichen Verhalten führen und Optimierungspotentiale durch die Wahl der Abarbeitungsreihenfolge bestehen.

5.3.1 Struktur der Evaluation

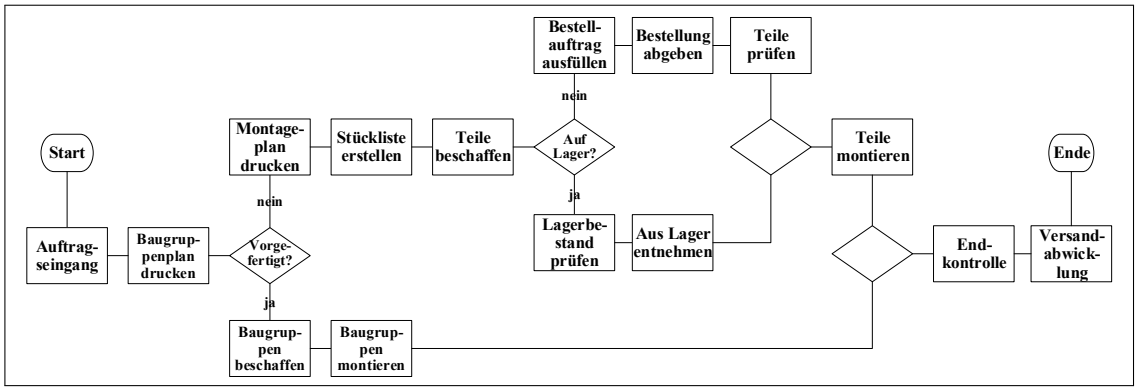
In diesem Kapitel wird die entwickelte Prozessarchitektur vorgestellt. Um die Simulation weitergehend zu erläutern, wird im Anhang für jede Aktivität die Ressourcennutzung aufgezeigt. Es wird erkennbar, dass sich einzelne Aktivitäten gemeinsame Ressourcen teilen, trotz der Tatsache, dass die Aktivitäten innerhalb unterschiedlicher Prozesse genutzt werden.

Für die Evaluation werden zuerst einzelne Prozesse herangezogen. Anhand zweier Prozesse wird gezeigt, dass sich Geschäftsprozesse gegenseitig negativ beeinflussen. Darauf folgend wird der Einfluss von Änderungen gezeigt. Für dieses Minimalbeispiel zeigt sich, dass selbst geringfügige Änderungen in einem Prozess Auswirkung auf andere Prozesse haben. Diese Änderungen können zum Beispiel das Hinzufügen weiterer Aktivitäten sein, um Sicherheitsziele einzuhalten, oder – um konform zur Gesetzgebungen zu sein – Änderungen im Zugriffskontext betreffen. Im letzten Evaluationsschritt soll mit diesen beiden Prozessen gezeigt werden, dass es die Simulation ermöglicht, die Abarbeitung der Aktivitäten so zu strukturieren, dass Wartezeiten auf Ressourcen minimiert werden und so die einzelnen Prozesse schneller beenden. Die Ergebnisse werden daraufhin anhand der gesamten Prozessarchitektur gezeigt.

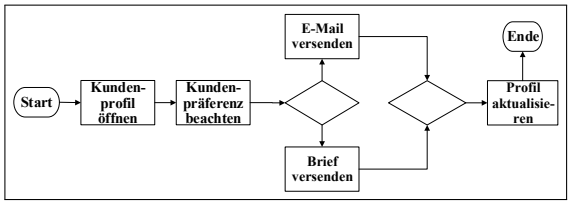
5.3.2 Simulierte Prozessarchitektur

Die zur Evaluation entwickelte Prozessarchitektur umfasst sieben Prozesse eines produzierenden Gewerbes. Diese sieben Prozesse sind in Abbildung 5.4 dargestellt. Zwei der Prozesse (5.4a sowie 5.4d) entspringen süddeutschen, mittelständischen Betrieben aus (Zahoransky et al. 2016) sowie (Zahoransky et al. 2014). Zusätzliche, unternehmenstypische Kernprozesse wie Schadensmeldung und regelmäßige Aufgaben wurden hinzugefügt. Die Prozesse umfassen dabei das Ausliefern einer Ware auf Bestellung, den Rechnungsein- und -ausgang, interne Bestellprozesse, die regelmäßige Sicherung der Kundendaten sowie Wartungsarbeiten und Kompensierung von Produktmängel. Diese Prozesse laufen unabhängig voneinander, teilen

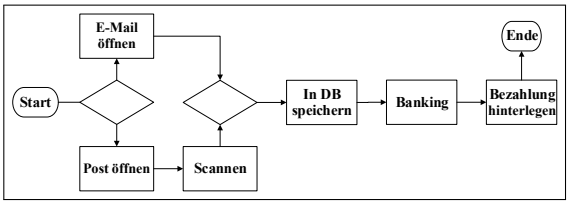
sich jedoch die im Unternehmen zu Verfügung stehenden Ressourcen, Betriebsmittel und Personal. Die Aktivität “Brief verfassen” teilt sich beispielsweise den Abteilungsdrucker mit den Wartungsaktivitäten des Prozesses in Abbildung 5.4f sowie mit Aktivitäten anderer Prozesse. Somit beeinflussen sich die eigentlich isolierten Prozesse untereinander, da die einzelnen Aktivitäten innerhalb der Prozesse in gegenseitiger Konkurrenz um Ressourcen stehen.



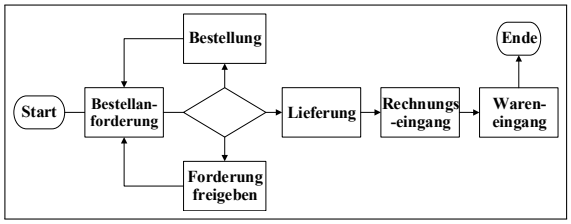
(a) Bestellungs und Produktionsprozess aus (Zahoransky et al. 2014)



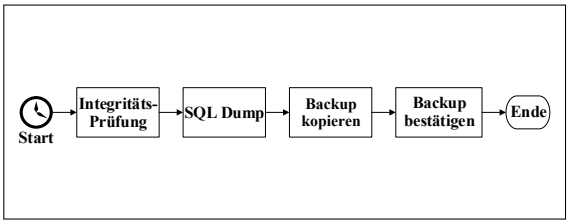
(b) Generischer Rechnungsausgangsprozess



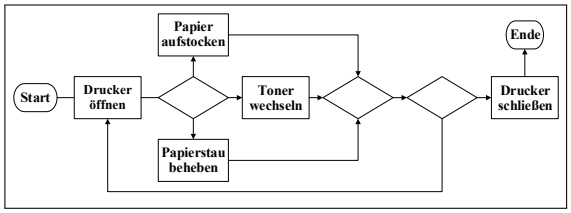
(c) Generischer Rechnungseingangsprozess.



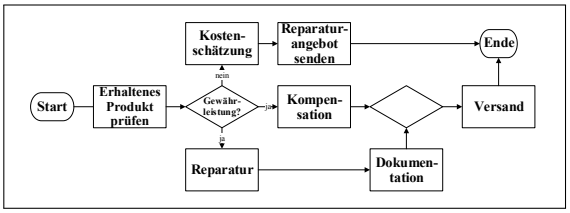
(d) Bestellprozess aus (Zahoransky et al. 2016)



(e) Regelmäßige Kopie der Stammdaten



(f) Wartungsarbeiten am Drucker



(g) Kompensierung

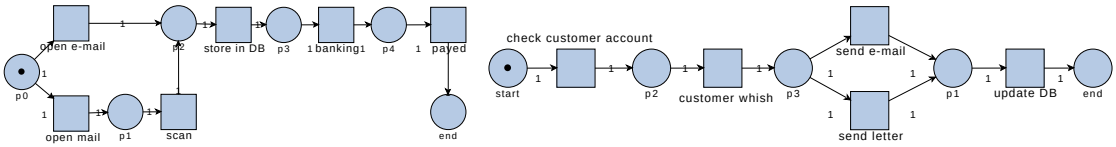
Abbildung 5.4.: Prozessarchitektur der Evalution

Die einzelnen Prozesse sind im Anhang B näher erläutert. Ebenfalls findet sich im Anhang die Auflistung der Ressourcen- und Betriebsmittelnutzung der Prozesse. Die aufgelistete Ressourcen- und Betriebsmittelnutzung ist lediglich auf Grund einer besseren Übersicht auf die einzelnen Prozesse aufgeteilt. Der Ressourcenkontext, welcher die abgebildeten Information beinhaltet, umfasst alle Tabellen und hat somit für alle Prozesse innerhalb der Prozessarchitektur Gültigkeit.

5.4 Gegenseitiger Einfluss von Prozessen

Eine Erweiterung der RTPN-Modelle gegenüber bestehenden Petri-Netzen ist auf Grund des geteilten Ressourcen-Kontexts und der geteilten Markierung M die Möglichkeit, gegenseitige Beeinflussungen von Prozessen aufzuzeigen.

Zur Evaluation dieser Fähigkeit werden die Rechnungsein- und Ausgangsprozesse 5.4b und 5.4c simuliert. Im Rahmen des Projekts GESINE (Eymann et al. 2014) wurden derartige Prozesse durch den ZUGFeRD (AWV e.V. 2014; Elter 2014) Standard zur automatisierten Rechnungsdurchführung digitalisiert. Die Prozesse können hiermit als Beispiel automatisierter Geschäftsprozessen gelten, deren Abarbeitung und Verhalten wohl definiert ist. Für die Simulation ist es nötig, die in BPMN hinterlegten Prozesse als RTP-Netz darzustellen. Das Ergebnis der manuellen Modellierung ist in den Abbildungen 5.5 erkennbar.

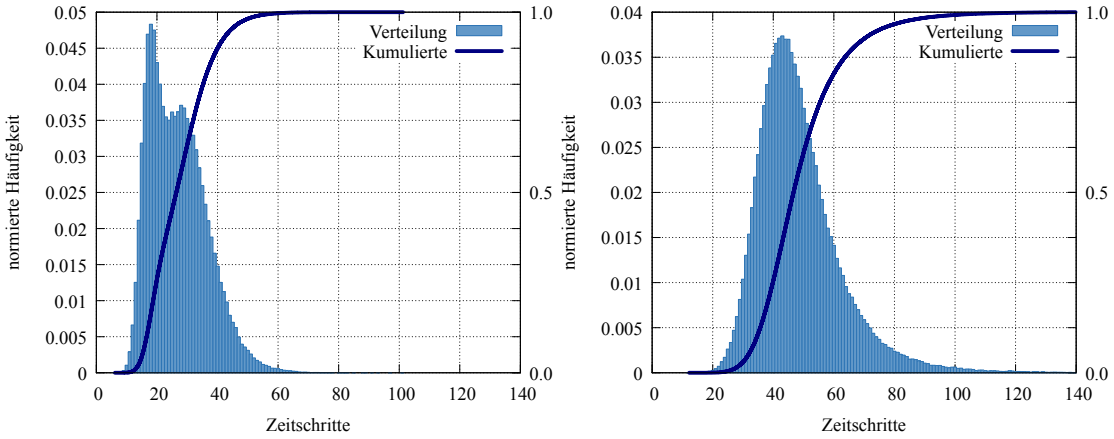


(a) Rechnungseingang als RTP-Netz (b) Rechnungsausgang als RTP-Netz

Abbildung 5.5.: Prozesse 5.4b und 5.4c als RTP-Netze

Zum Beispiel wird gefordert, dass im Durchschnitt keiner der Prozesse länger als 60 Zeiteinheiten benötigen soll, um fristgerecht und somit wirtschaftlich zu sein. Eine Einzelsimulation der Prozesse zeigt, dass dieses Ziel in über dreiviertel der Fälle für jeden Prozess einzeln eingehalten wird. Die Ergebnisse dieser Einzelsimulationen sind in Abbildung 5.6 zu erkennen. Als helle Fläche ist die Wahrscheinlichkeitsdichte zu erkennen, dass der Prozess zu einem gewissen Zeitpunkt endet. Für Abbildung 5.6b zeigt sich, dass der simulierte Prozess ungefähr bei Zeitschritt

20 die höchste Wahrscheinlichkeit aufweist zu enden. Die durchgehende Linie zeigt die Gesamtwahrscheinlichkeit, dass der Prozess bis zu einem Zeitschritt x endet.



(a) Rechnungseingangsprozess (b) Rechnungsausgangsprozess

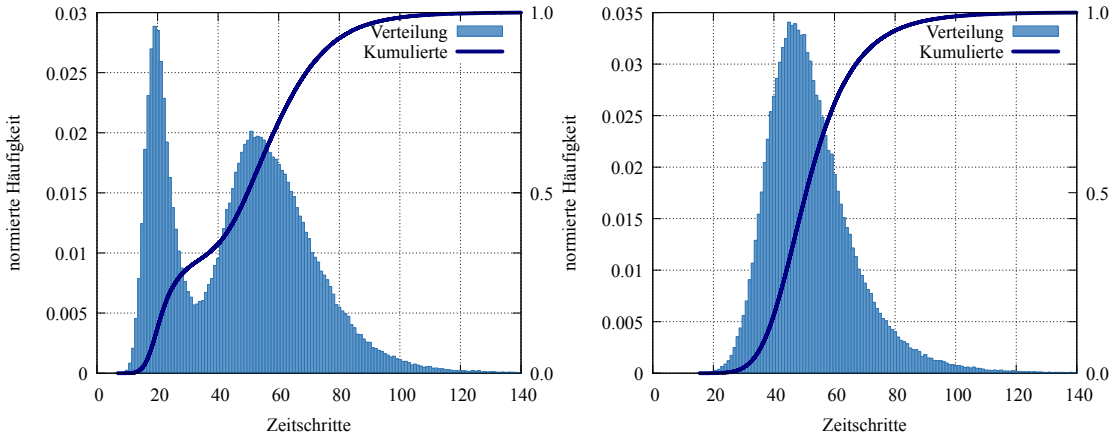
Abbildung 5.6.: Einzelsimulationen der Rechnungsein- und ausgangsprozesse

Einzelsimulationen sind bereits mit bestehenden Simulationen und Modellierungen möglich, da in diesem Fall die Ressourcennutzung nicht behandelt wird. Hier stehen alle Ressourcen dem einen, simulierten Prozess zur Verfügung. Der Prozess muss nicht mit anderen Prozessen um die Ressourcennutzung konkurrieren.

Diese Ergebnisse verlieren jedoch ihre Gültigkeit, wenn die Prozesse nicht mehr isoliert abgearbeitet werden und die Ressourcen unter den Prozessen geteilt werden. Diese Tatsache lässt sich durch die simultane Simulation belegen. Solch eine simultane Simulation ist durch bestehende Modelle, wie die Timed Petri-Nets jedoch nicht möglich. TdPNs und anderen Petri-Netz-basierten Modellierungen fehlen die Fähigkeit, Ressourcen so zu beschreiben, dass sie von mehreren Prozessen gleichzeitig aus nutzbar sind. Die Konkurrenz um Ressourcen unter verschiedenen Prozessen ist damit mit bestehenden Modellen nicht abbildbar.

Die Ergebnisse in Abbildung 5.7 zeigen, dass die Erfolgswahrscheinlichkeit beider Prozesse verringert ist, wenn diese gleichzeitig innerhalb einer Prozessarchitektur ausgeführt werden.

Für den Rechnungsausgangsprozess verringert sich die Erfolgswahrscheinlichkeit, nach spätestens 60 Zeitschritten vollendet zu sein, von 82,7 % auf 74,2 %.



(a) Rechnungseingangsprozess (b) Rechnungsausgangsprozess

Abbildung 5.7.: Simultane Simulation der Rechnungsein- und ausgangsprozesse

Für den Rechnungseingangsprozess betrug die ursprüngliche Erfolgsrate im Falle einer isolierten Ausführung 99,7 %. Diese fällt durch die simultane Ausführung auf 69,9 %.

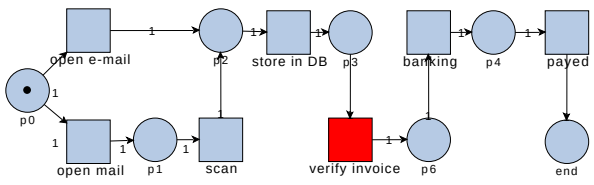
Die kombinierte, gemittelte Erfolgswahrscheinlichkeit beider Prozesse sank von 91,2 % auf 66,6 %. Somit zeigt sich, dass einzelne Prozesse sehr empfindlich auf Schwankungen der Verfügbarkeit ihrer Ressourcen reagieren können.

5.5 Einfluss sicherheitsbezogener Kontrollflussänderungen

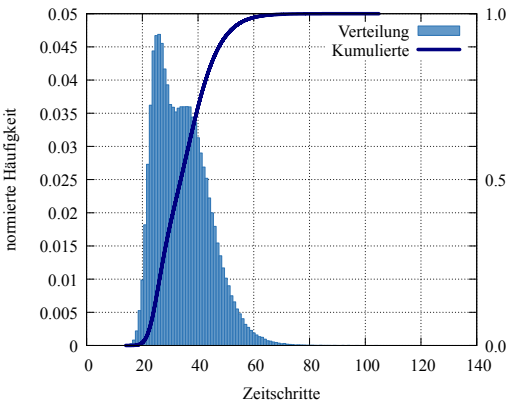
Sicherheitsrelevante Bedenken, wie die Einhaltung von Compliance, interne oder externe Sicherheitsvorschriften sowie die Einhaltung von Gesetzen kann ein Unternehmen dazu zwingen, bestehende Prozesse anzupassen. Diese Veränderungen können sich auf einzelne Aktivitäten beziehen oder führen dazu, dass neue Aktivitäten zu bestehenden Prozessen hinzukommen oder geändert werden. Je nach Anforderung kann die Anpassung von Prozessen an neue Bedürfnisse auch eine umfassende Neuentwicklung bedeuten. Die Auswirkungen dieser Änderungen betreffen jedoch nicht zwingend nur den angepassten Prozess. Änderungen eines Prozesses können zum Beispiel bewirken, dass dieser mehr Ressourcen benötigt oder dass sich durch Änderungen der Zugangskontrolle die zur Durchführung nötigen Personen ändern. Diese Änderungen können die Ausführung andere Prozesse stören. Es ist auch möglich, dass die Änderungen das zeitliche Verhalten beeinflussen und dadurch

andere Prozesse länger auf Ergebnisse warten müssen. In vielen Fällen lassen sich die Auswirkungen, die durch Änderung eines Prozesses auf die Prozessarchitektur entstehen, bisher nicht a priori abschätzen (Browning et al. 2002). Für Unternehmen ist es jedoch notwendig, ihre Prozesse so zu gestalten, dass sowohl die Kosten, als auch die Zeit für einzelnen Prozesse optimal sind (Browning 2003). Hierfür ist eine Abschätzung der gegenseitige Beeinflussung hilfreich.

Für eine Demonstration, wie sich Änderungen eines Prozesses auf andere Prozesse auswirken, wird der Rechnungseingangsprozess so verändert, dass eingehende Rechnungen zuerst geprüft werden, bevor eine Bezahlung stattfindet. Diese zusätzliche Aktivität innerhalb des Prozesses soll Betrug erschweren und somit Risiken von dem Unternehmen fern halten und könnte Teil einer Compliance-Maßnahme darstellen, um die Rechtmäßigkeit der Buchhaltung zu gewährleisten. Die Änderungen des Prozesses sind in Abbildung 5.8a abgebildet. Diese besteht darin, dass eine weitere Aktivität, die Rechnungsprüfung eingefügt wurde. Die Prüfung einer Rechnung benötigt im Schnitt 7 Zeiteinheiten. Die Auswirkungen auf den einzelnen Prozess selbst sind vernachlässigbar. Statt einer isolierten Erfolgsquote von 99 % beträgt diese nun 98 %. Die Ergebnisse der isolierten Simulation sind in Abbildung 5.8b erkennbar dargestellt.



(a) Änderung am Rechnungseingangsprozess

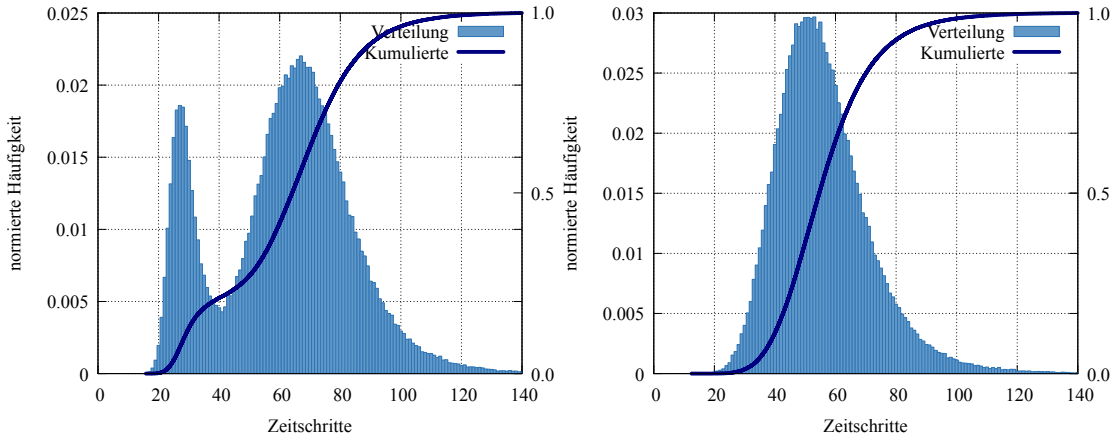


(b) Einzelsimulation des geänderten Prozesses

Abbildung 5.8.: Änderungen des Prozesses und zugehöriges Simulationsergebnis

Diese geringfügige Veränderung scheint ein akzeptabler Kompromiss zwischen Sicherheit und Aufwand zu sein, vor allem wenn die dadurch minimierten Risiken betrachtet werden. Der veränderte Prozess fügt sich jedoch nicht mehr wie gewohnt in die bestehende Prozessarchitektur ein. Da innerhalb der Prüfaktivität

Personal gebunden ist, reserviert der Rechnungseingangsprozess bestehende Ressourcen länger. Werden der veränderte Rechnungseingangsprozess zusammen mit dem Rechnungsausgangsprozess simuliert, ergibt sich eine große Änderung gegenüber den ursprünglichen Ergebnissen, welche in den Grafiken in Abbildungen 5.9 zu erkennen sind. Der geänderte Rechnungseingang (Grafik in 5.9a) hat nun nur noch eine 41,5 % Chance, dass er innerhalb der geforderten 60 Zeiteinheiten vollendet ist. Im Falle des Rechnungsausgangs (Grafik in 5.9b) sind es im Durchschnitt nur noch 64,5 % der Prozesse, die rechtzeitig fertig werden. Die durchschnittliche Erfolgsrate beider Prozesse sank damit von ursprünglich 66,6 % auf nur noch 53 %. Dies zeigt, dass strukturelle Änderungen eines Prozesses andere, von den gleichen Ressourcen abhängige Prozesse, beeinflussen können. In diesem Fall zeigt die simultane Simulation der Prozesse durch die RTPN-Modelle die Beeinträchtigung der Verfügbarkeit durch die Gesetzeskonformität.



(a) Prozesssimulation des geänderten Rechnungseingang

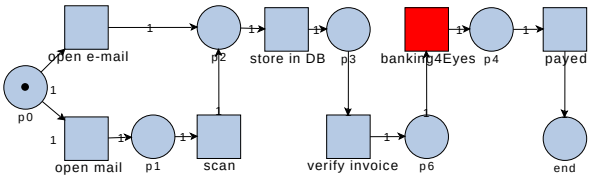
(b) Simulation des Rechnungsausgangs

Abbildung 5.9.: Prozessarchitektursimulation nach Änderungen am Rechnungseingang durch zusätzliche Prüfaktivität

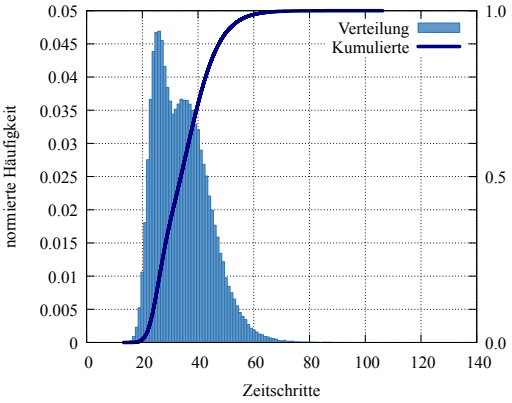
5.6 Einfluss durch Rechteverwaltung

Änderungen durch unterschiedliche Rechtezuweisungen können ebenfalls zu einem verändertem Verhalten der Prozesse führen. So könnte ein Unternehmen darauf angewiesen sein, die Bezahlung der Rechnungen durch zwei Personen durchführen zu lassen, damit es zu keinen Betrugsfällen kommt. Dadurch bindet eine Aktivität zwei Personen, die nicht mehr für andere Aktivitäten zur Verfügung stehen

können oder die Aktivität selbst muss länger auf die Verfügbarkeit der notwendigen Mitarbeitergruppen warten. Im zuvor genannten Beispiel würde dies dazu führen, dass die Aktivität “Banking”, also die Überweisung des Geldbetrags, durch zwei Personen stattfindet. Die Änderungen des Prozesses sind in Abbildung 5.10a markiert.



(a) Änderung des Rechnungseingangsprozesses durch Vier-Augen Prinzip



(b) Simulationsergebnis der Einzelsimulation mit dem Vier-Augen Prinzip

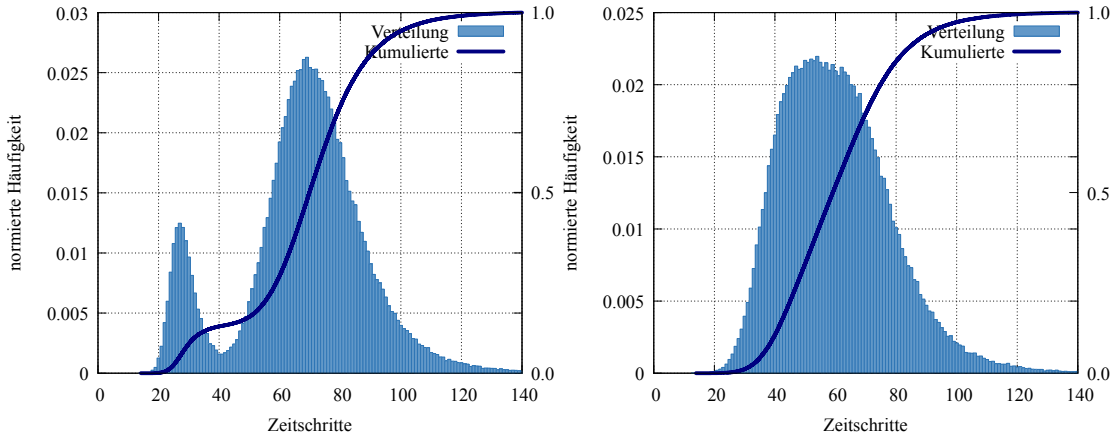
Abbildung 5.10.: Rechnungseingangsprozess mit Sicherheitsbestimmungen

Eine Einzelsimulation dieses Prozesses (Abbildungen 5.10) zeigt keine Änderung im Zeitverhalten, wie im Simulationsergebnis 5.10b erkennbar ist. Die Erfolgschance des Prozesses, nach 60 Zeiteinheiten beendet zu sein, liegt bei 98,9 %.

Die Nachteile der Rechteänderung zeigen sich in dem Zusammenspiel mit anderen Prozessen, wie in den Ergebnissen in Abbildungen 5.11 zu erkennen ist.

Der geänderte Rechnungseingangsprozess zeigt nun nur noch eine Erfolgschance von 27,2 %, dass er rechtzeitig zu einem Ende kommt. Zusätzlich zeigt auch der Rechnungsausgangsprozess eine Verschlechterung. Diesem wurden durch die Vier-Augen Bestimmung des Rechnungseingangsprozesses Mitarbeiter entzogen, die nun nicht mehr ihre Arbeit auf den Rechnungsausgangsprozesses konzentrieren können. So fiel die Erfolgschance der rechtzeitigen Terminierung des Rechnungsausgangsprozesses auf 51,6 %. Damit sinkt die gemittelte Erfolgswahrscheinlichkeit der Prozesse von 53 % auf 39,3 %.

Hiermiet beweisen die RTPN-Modelle eine weitere Stärke ihrer Ausdrucksfähigkeit, indem sie die Auswirkungen der Änderungen einer Rechtezuweisung innerhalb einer Prozessarchitektur quantifizierbar machen.



(a) Zeitverhalten des Rechnungseingangs-prozesses mit Vier-Augen Prinzip.

(b) Zeitverhalten des Rechnungsausgangsprozesses.

Abbildung 5.11.: Simultane Simulation des Rechnungseingangsprozesses mit zusätzlichen Sicherheitsbestimmungen und des Rechnungsausgangsprozesses

5.7 Kosten von Sicherheit

Die Simulationsergebnisse der Zeitdauern der einzelnen Prozesse lassen sich heranziehen, um eine Kostenschätzung der Prozesse durchzuführen. Eine hierfür angewandte Methode der Prozesskostenrechnung ist das TD ABC (Cooper 1988; Özyürek et al. 2014; Kaplan et al. 2004). Diese Methode ordnet den Prozessen eine von der Zeit abhängige Kostenfunktion zu. Die Ermittlung der Prozessdurchlaufzeiten lässt somit Rückschlüsse auf die verursachten Kosten zu. Zumeist werden diese Kosten durch die notwendige Personalbindung quantifiziert, da die Personalkosten maßgeblich sind (Werner 2013).

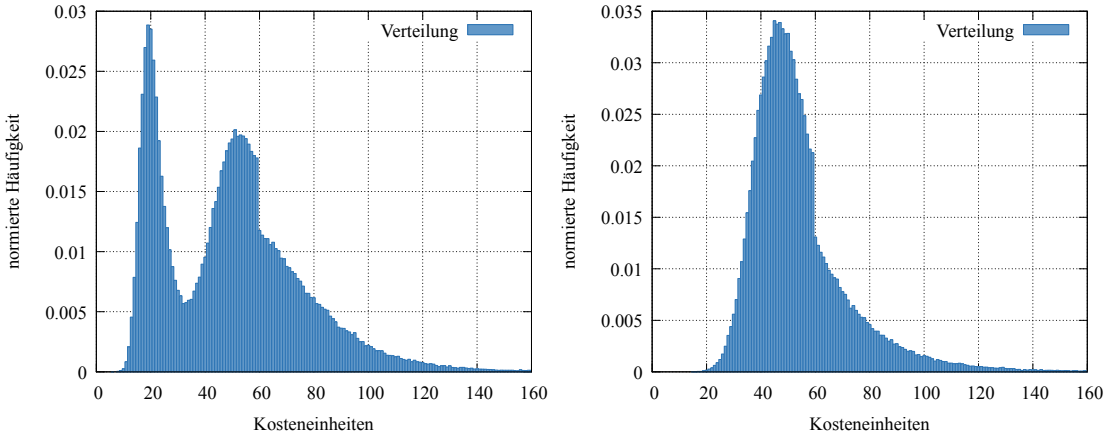
Die zeitliche Simulation der Prozesse lässt eine exakte Ermittlung dieser Kosten zu, da für jede Aktivität die simulierte Zeitdauer und Personalbindung bekannt ist. Statt die Kostenberechnung anhand von Durchschnittswerten durchzuführen, können durch die Simulation detaillierter Ergebnisse erzielt werden.

In den folgenden Beispielen wird der Einfachheit halber eine Kosteneinheit pro Zeiteinheit und Prozess angesetzt. Dies gilt, so lange die Prozesse innerhalb ihrer geplanten Zeitdauer terminieren. Nach der Überschreitung der geplanten Zeitdauer können Nachzahlungen oder Mahnungen anfallen oder Einnahmen zu spät reali-

siert werden. Für jede weitere Zeiteinheit des Prozesses, die länger als geplant dauert, werden daher Kosten von 1,5 Einheiten pro Zeiteinheit angesetzt.

5.7.1 Ausgangspunkt der Prozesskostensimulation

Für eine beispielhafte Simulation der Kosten dienen die bereits gezeigten Prozesse des Rechnungseingangs, als auch des Kostenausgangs. Als Ausgangspunkt dienen die unveränderten Prozesse, siehe Abbildung 5.5, ohne zusätzliche Prüfaktivität oder Vier-Augen-Richtlinie.



(a) Kostensimulation des Rechnungseingangsprozesses

(b) Kostensimulation des Rechnungsausgangsprozesses

Abbildung 5.12.: Simultane Kostensimulation der unveränderten Rechnungsein- und ausgangsprozesse.

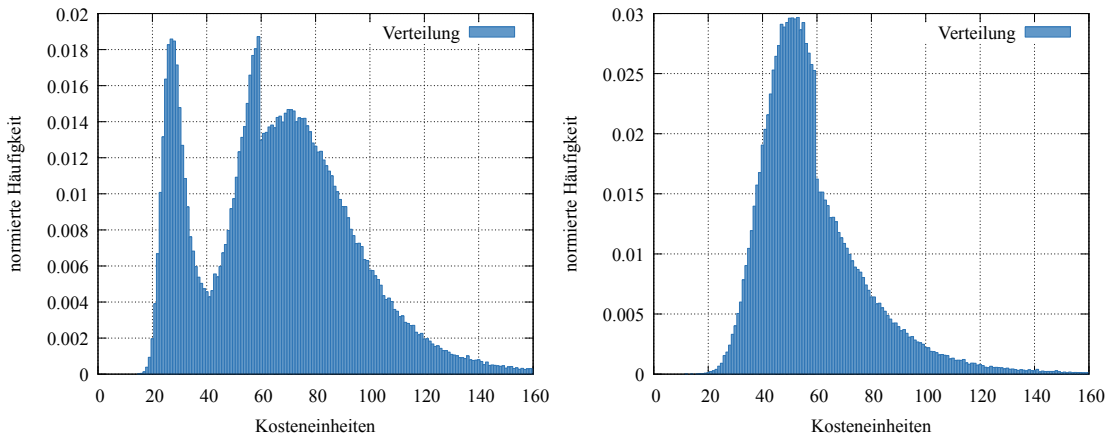
Die Ergebnisse der Kostensimulation sind in den Abbildungen 5.12 hinterlegt. In diesen und den folgenden Abbildungen zeigt das Histogramm nicht mehr die Verteilung der Abarbeitungszeiten, sondern die Wahrscheinlichkeitsverteilung der verursachten Kosten. Die Kostensimulation zeigt für den Rechnungseingangsprozess durchschnittliche Kosten von 50,1 (Abbildung 5.12a). Für den Rechnungsausgangsprozess liegen die durchschnittlichen Kosten bei 54,8 Einheiten.

Die Verteilung der Kosten zeigt ab 60 Kosteneinheiten einen abrupten Einbruch. Dieser entsteht durch die Überschreitung der festgelegten Abarbeitungszeit von 60 Zeiteinheiten. Bis zu diesem Punkt entspricht die Art der Verteilung mit einer Kosteneinheit pro Zeitschritt der Verteilung aus den Zeitsimulationen. Ab 60

Zeiteinheiten sind jedoch Kosten von 1,5 Einheiten pro Zeiteinheit angenommen, weswegen sich die Verteilung streckt.

5.7.2 Kosten durch zusätzliche Prüfaktivität

In Abschnitt 5.5 wurde der Rechnungseingang so angepasst, dass Rechnungen vor dem Bezahlvorgang nochmals auf ihre Richtigkeit geprüft werden. Die Ergebnisse zeigten negative Einflüsse auf das zeitliche Verhalten beider Prozesse. Die monetären Auswirkungen dieser Änderung spiegeln sich verstärkt in der Kostensimulation wieder, da nun der geplante Terminierungszeitpunkt nach 60 Zeiteinheiten in weniger Fällen erreicht wird.



(a) Kostensimulation des geänderten Rechnungseingangsprozesses

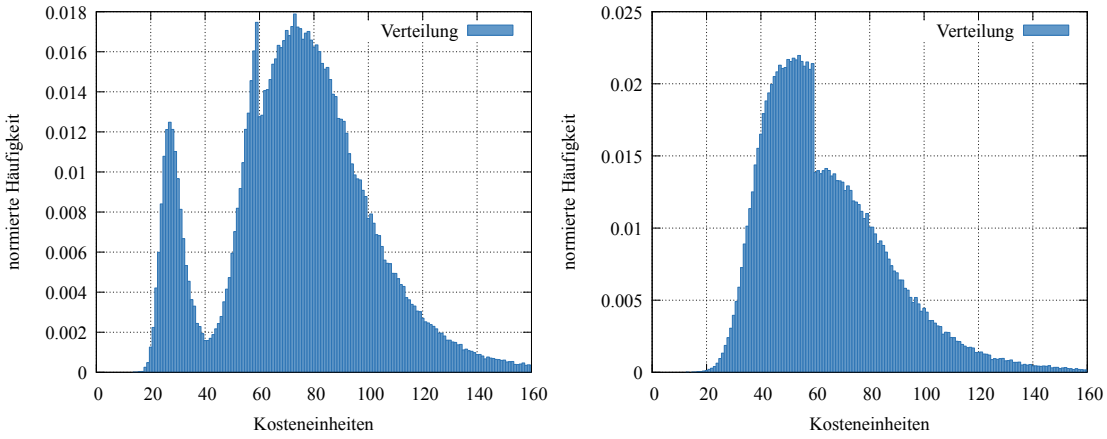
(b) Kostensimulation des Rechnungsausgangsprozesses

Abbildung 5.13.: Simultane Kostensimulation der Rechnungsein- und ausgangsprozesse mit zusätzlicher Prüfaktivität.

Die Ergebnisse der Kostensimulation in Abbildungen 5.13 zeigen, dass die Prozesskosten sich durch die zusätzliche Aktivität weiter erhöhen. Die durchschnittlichen Kosten der Prozessausführung des Rechnungseingangsprozesses belaufen sich mit der Prüfaktivität auf 67,4 Einheiten. Da der Rechnungsausgangsprozess vermehrt auf Ressourcen oder Personal warten muss, steigen auch die Kosten dieses Prozesses auf 59,2 Kosteneinheiten. Die Anteile der Kosten über 60 Einheiten zeigen sich vermehrt in den Histogrammen 5.13a und 5.13b.

5.7.3 Kosten durch Änderung der Zugriffskontrolle

Als letztes Beispiel wurden ein verändertes Zeitverhalten im Falle einer Änderung der Rechteverwaltung gezeigt. Eine Aktivität im Rechnungseingangsprozess wurde so verändert, dass sie durch zwei Personen bearbeitet werden muss. Diese Bindung von Personal zeigte ebenfalls Auswirkungen auf beide Prozesse.



(a) Kostensimulation des geänderten Rechnungseingangsprozesses

(b) Kostensimulation des Rechnungsausgangsprozesses

Abbildung 5.14.: Simultane Kostensimulation der Rechnungsein- und ausgangsprozesse mit Vier-Augen Prinzip aus 5.10a

Die durchschnittlichen Kosten des Rechnungseingangsprozesses steigen durch die Anwendung des Vier-Augen-Prinzips auf 75 Einheiten. Da auch der Rechnungsausgangsprozess durch die Personalbindung betroffen ist, steigen auch dessen Kosten auf 75,2.

5.8 Optimierende Steuerung einer Prozessarchitektur

Die Simulation zeigt die Einflüsse von Sicherheits- und Complianceänderungen auf das Zeitverhalten, sowie die Kosten einer Prozessarchitektur unter Beachtung der Ressourcen- und Personalabhängigkeit der einzelnen Prozesse. Die Simulation generiert in jedem Durchlauf randomisierte Abarbeitungsreihenfolgen der Aktivitäten, welche von der Markierungsmenge M vorgehalten werden. Ein Vergleich

dieser Einträge miteinander erlaubt die Optimierung von Abarbeitungsreihenfolgen entsprechend der Gewichtung der Geschäftsprozesse oder den Kosten (siehe Kapitel 4.6). Unterschiedliche Abarbeitungsreihenfolgen zeigen unterschiedliche Ressourcennutzung. Eine günstige Ressourcennutzung führt dazu, dass Aktivitäten nicht auf Ressourcen warten müssen und ohne Beeinflussung durch andere Prozesse parallel abgearbeitet werden können. Die Ergebnisse der Optimierung erlauben somit eine effiziente Steuerung der Prozessarchitektur, um die Wartezeiten auf Ressourcen zu verringern oder stark gewichteten Prozessen vorrangigen Zugriff auf Ressourcen zu gewähren. Gerade die Einhaltung von Gesetzesnormen kann Prozessänderungen fordern, welche sowohl unumgänglich sind als auch die Verfügbarkeit der Prozessarchitektur stören. Eine optimierende Prozesssteuerung bietet die Möglichkeit, diese negativen Auswirkungen zu mildern.

Die Optimierung einer Prozessarchitektur wird anhand der Rechnungsein- und Ausgangsprozesse und daraufhin auf die gesamten Prozessarchitektur aus Abbildung 5.4 gezeigt.

5.8.1 Beispiel

In der Evaluation wurde gezeigt, dass beide Prozesse durch die zusätzliche Prüfaktivität und das Vier-Augen Prinzip gestört wurden und ein schlechtes Zeitverhalten zeigten. Ein Ablauf der Aktivitäten beider Prozesse ist in Abbildung 5.15 grafisch aufgezeigt.

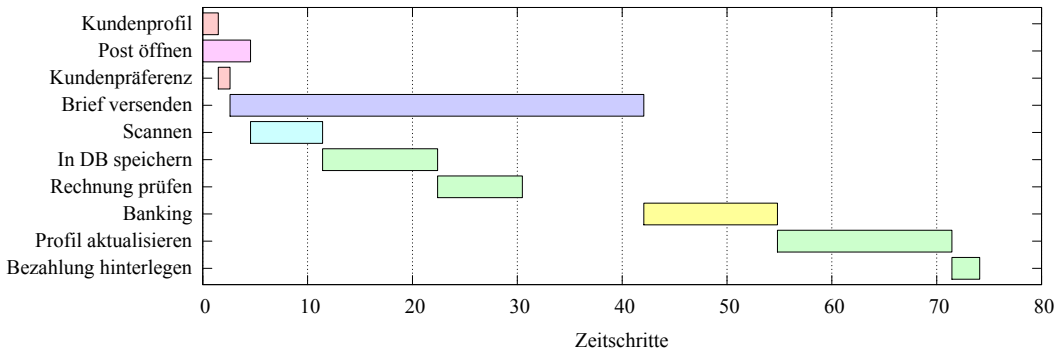


Abbildung 5.15.: Aktivitätenabfolge der Rechnungsein- und Ausgangsprozesse

Die Abbildung zeigt einen Ablaufplan mit langen Aktivitäten. Statt die Korrespondenz elektronisch durchzuführen, wird sie in diesem Ablaufplan postalisch

umgesetzt. Das Resultat ist, dass die Ausführung einer Banküberweisung lange aufgeschoben werden muss, da die langwierige Korrespondenz Personal bindet. Die Aktivitäten beider Prozesse sind voneinander abhängig, da durch das geforderte Vier-Augen-Prinzip jeweils eine Person aus den unterschiedlichen Rollen “Sekretariat” und “Ein- und Verkauf” benötigt wird. Die Abarbeitung der Aktivitäten im Rechnungsausgangsprozess stört die Abarbeitung des Rechnungseingangs. Diese Personal- und Ressourcennutzung ist in Abbildung 5.16 zu erkennen.

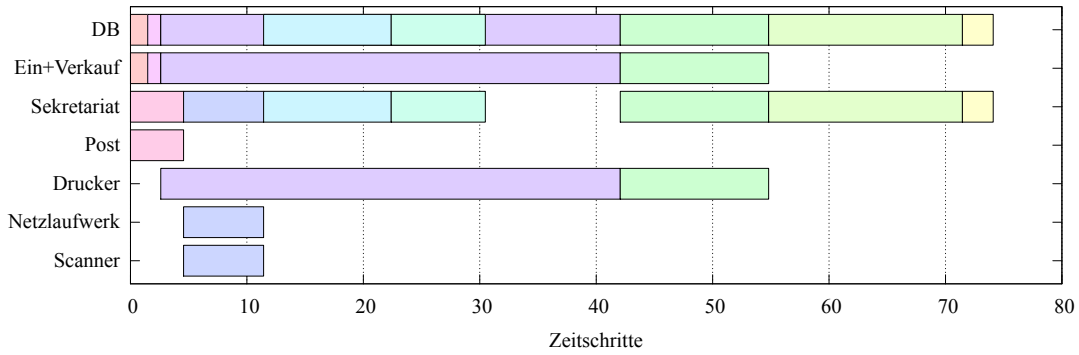


Abbildung 5.16.: Ressourcennutzung der Rechnungsein- und Ausgangsprozesse

Die Ressourcennutzung zeigt Lücken in der Auslastung des Sekretariats, da die zugehörigen Mitarbeiter nicht alleine mit der Banküberweisung fortfahren können, sondern darauf warten müssen, dass Personal aus der Rolle “Ein- und Verkauf” frei wird.

Die Simulationsergebnisse für diese Abarbeitungspfade zeigen eine gemittelte Erfolgswahrscheinlichkeit von 37,1 %. Die durchschnittlich entstehenden Kosten belaufen sich in diesem Fall auf 138,85 Einheiten.

Eine Optimierung basierend auf der Simulationen unterschiedlicher Abarbeitungsreihenfolgen zeigt, dass ein elektronischer Versand der Rechnung im Mittel schneller ist und somit die Aktivitäten des Rechnungseingangsprozesses früher beginnen können. Ein solche optimierter Abarbeitungsplan ist in Abbildung 5.17 abgebildet. Es zeigt sich, dass die Simulation kürzere Aktivitäten bevorzugt.

Die verkürzte Korrespondenz erlaubt die frühe Zusammenarbeit beider Rollen, sodass die Überweisungsaktivität zu einem früherem Zeitpunkt beginnen kann, siehe Abbildung 5.18.

Mit dieser Abarbeitungsreihenfolge zeigt die Simulation eine Erfolgswahrscheinlichkeit von 75,5 %, dass beide Prozesse innerhalb ihrer geforderten Frist enden.

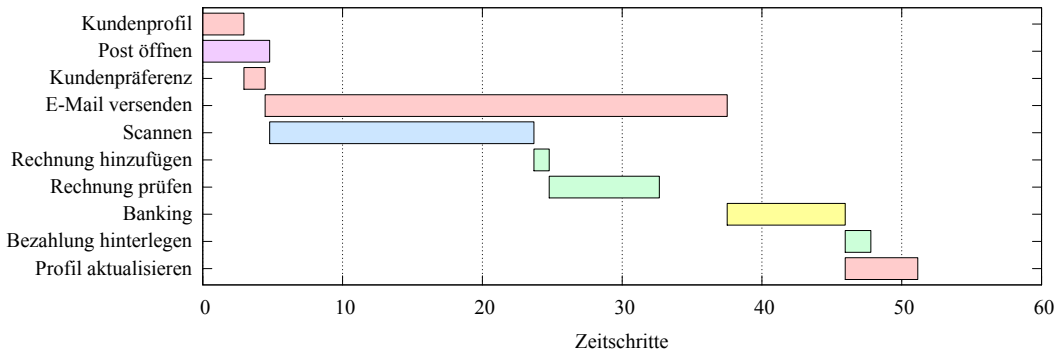


Abbildung 5.17.: Optimierte Aktivitätenabfolge der Rechnungsein- und ausgangsprozesse

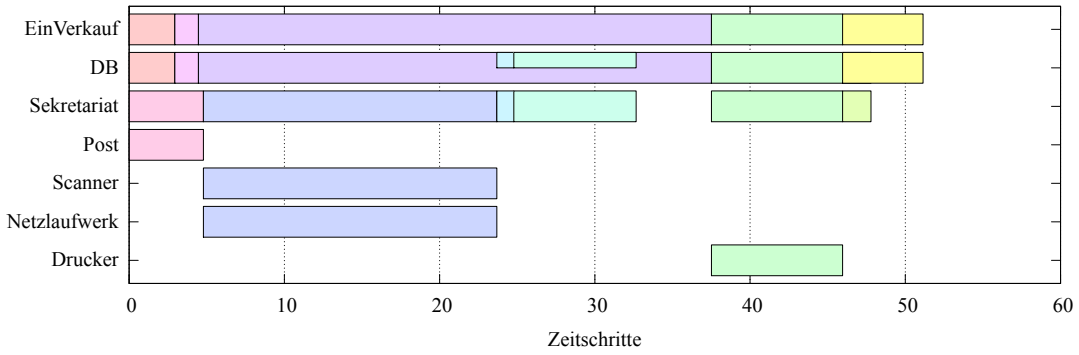
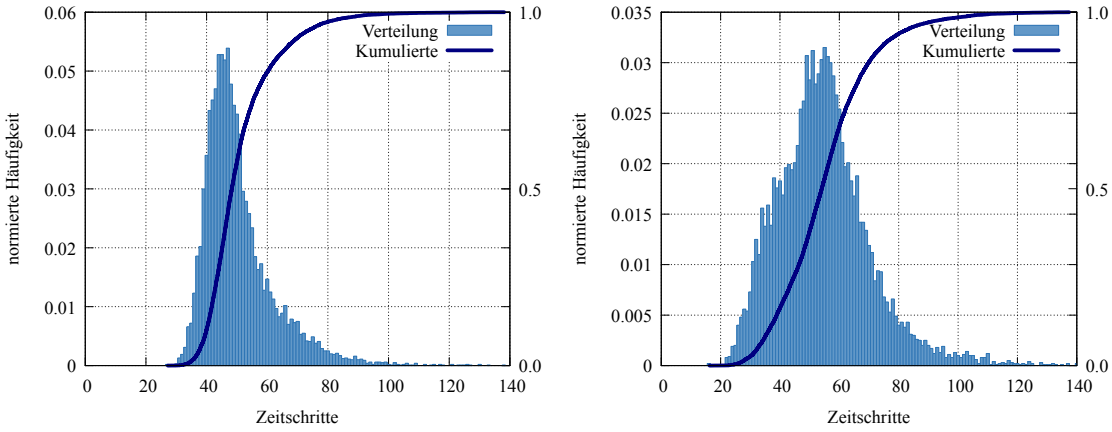


Abbildung 5.18.: Optimierte Ressourcennutzung der Rechnungsein- und ausgangsprozesse

Die führt zu durchschnittlichen Kosten von 109,3 Einheiten. Die Ergebnisse der Simulation dieser Abarbeitungsreihenfolge sind in den Abbildungen 5.19 erkennbar. Im Vergleich zu der ungesteuerten Abarbeitung in Abbildungen 5.11 wird das verbesserte Zeitverhalten erkennbar, welches sich auch auf die zu erwartenden Kosten auswirkt.

5.8.2 Optimierung nach Gewichtung der Prozesse

Auch größere Prozessarchitekturen, deren Prozesse unterschiedliche Abarbeitungspfade beinhalten, lassen sich wie gezeigt optimieren. Das Optimierungsziel kann eine kurze Abarbeitungszeit, wie im Beispiel zuvor, sein. Es ist aber auch möglich, entsprechend der Prozessgewichtung die Prozessarchitektur so zu steuern, dass die gewichteten Prozesse möglichst störungsfrei abgearbeitet werden. Ein weiteres Kriterium kann die Minimierung der entstandenen Kosten sein.



(a) Optimiertes Zeitverhalten des Rechnungseingangs mit Prüfaktivität und Vier-Augen Prinzip

(b) Optimiertes Zeitverhalten des Rechnungsausgangsprozesses

Abbildung 5.19.: Optimiertes Zeitverhalten der Rechnungsein- und ausgangsprozesse

Die Simulation der gesamten Prozessarchitektur zeigt, dass in durchschnittlich 61,32 % der Fälle die Prozesse fristgerecht enden und zusammen Kosten in Höhe von 1322,35 Einheiten verursachen. In den folgenden Ergebnissen wird der Produktionsprozess doppelt gewichtet, da er im Beispiel maßgeblich zur Wertschöpfung beiträgt. Das Zeitverhalten des Produktionsprozesses ist in Abbildung 5.22a erkennbar. Hier zeigen sich die unterschiedlich langen Ablaufpfade des Prozesses. Sind für ein bestelltes Produkt bereits Baugruppen vorhanden, können diese montiert und versandt werden. Falls die Baugruppen jedoch nicht vormontiert sind, benötigt der Prozess mehr Zeit, da Einzelteile zuerst bestellt werden müssen. Dies erklärt die beiden zeitlichen Häufungen innerhalb der Statistik.

In Abbildung 5.20 wird eine mögliche, zufällige Abarbeitungsreihenfolge der Aktivitäten der Prozessarchitektur gezeigt. In dieser Abarbeitung erfolgt die Auslieferung eines Produktes verzögert, da Einzelteile bestellt werden. Das Ergebnis ist eine durchschnittliche, fristgerechte Abarbeitung in 19,78 % der Fälle und durchschnittliche Kosten von 3124,73 Einheiten.

Die automatisierte Optimierung der Prozessarchitektur ermittelt sowohl schnelle Pfade innerhalb der Prozesse und reduziert die Wartezeit der Aktivitäten auf Ressourcen. Abbildung 5.21 zeigt eine alternative Abarbeitungsreihenfolge der Prozessarchitektur, welche kürzere Pfade der Einzelprozesse bevorzugt, als auch zeitlich lange Aktivitäten frühzeitig startet, damit die Prozess innerhalb ihrer

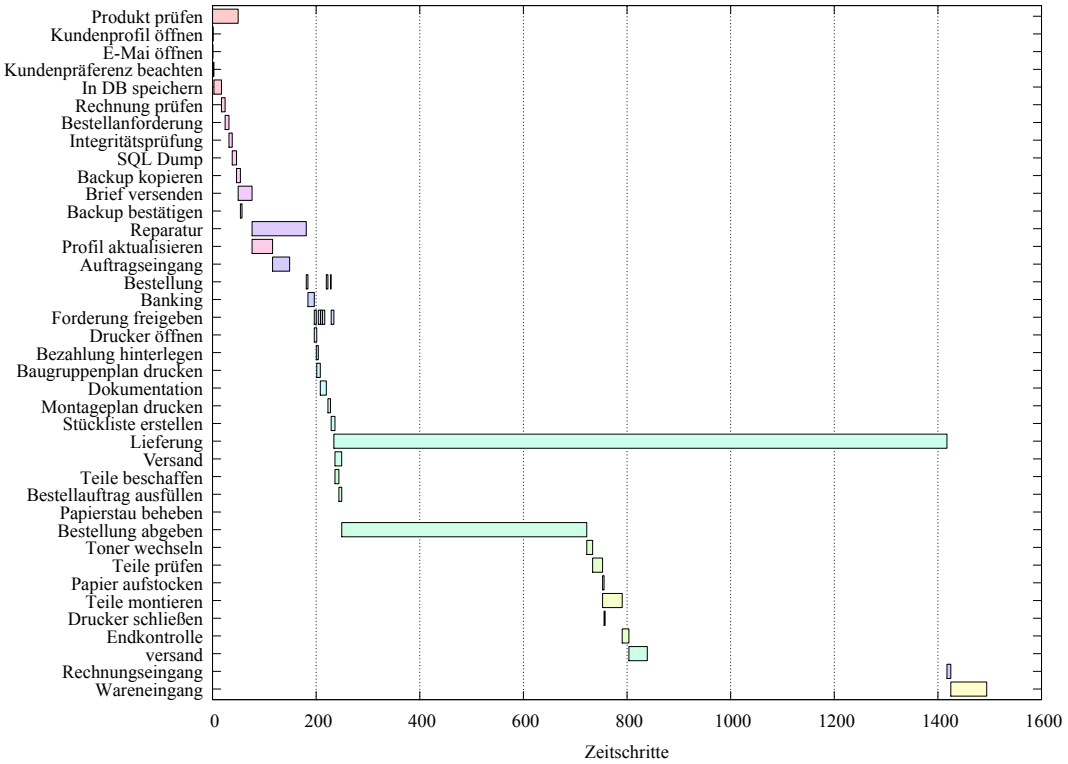


Abbildung 5.20.: Abarbeitung der Prozessarchitektur

Frist enden können. Die Optimierung erfolgte entsprechend der doppelten Gewichtung des Produktionsprozesses. Dieser optimierte Ablaufplan zeigt eine Einhaltung der Fristen in 75,9 % der Fälle und erzeugt durchschnittliche Kosten von 861,71 Einheiten.

Der Einfluss des verbesserten Ablaufplans auf den Produktionsprozess ist in Abbildung 5.22 dargestellt. Sie zeigt, dass der Produktionsprozess ohne Störungen arbeitet und aufgrund der kürzeren Ablaufpfade schneller zu einem Endergebnis führt.

5.9 Zusammenfassung

In der Evaluation wurden die Implementierung der RTP Nets und deren Einsatzmöglichkeiten gezeigt. Die Implementierung erfolgte innerhalb der Softwareplattform SWAT, da sich diese wegen den bestehenden Netzdefinitionen für Petri-Netze eignet. Sowohl die Architektur als auch die wichtigsten Methoden der implemen-

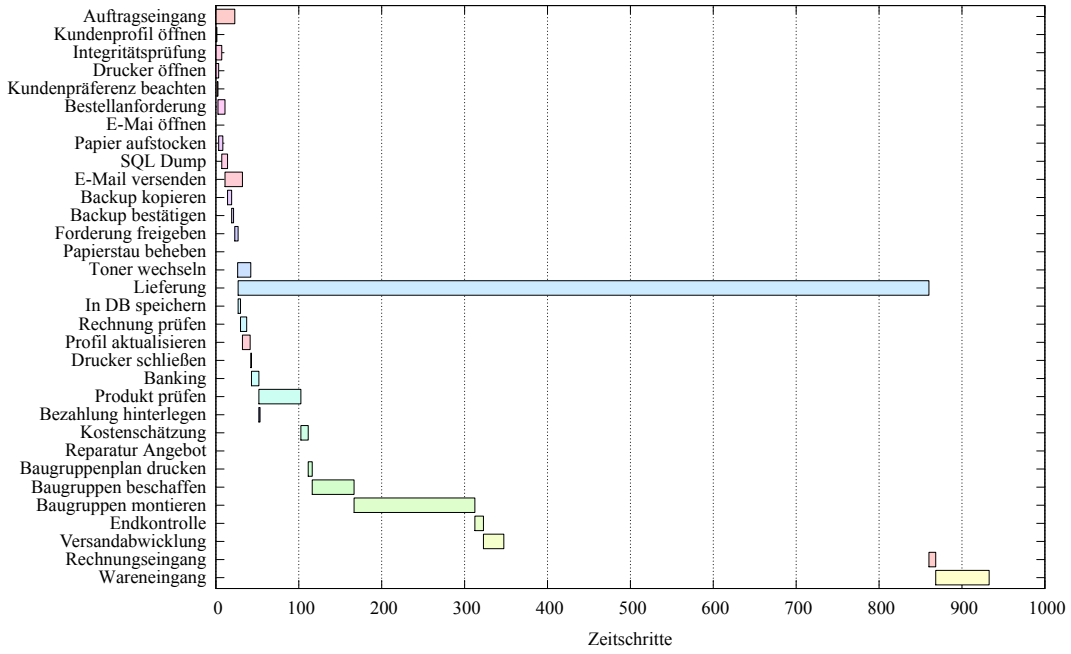
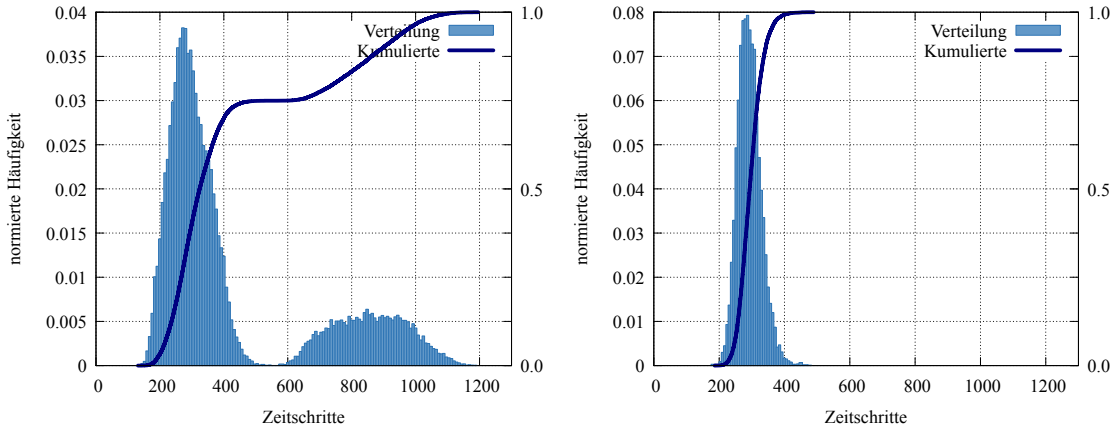


Abbildung 5.21.: Optimierte Abarbeitung der Prozessarchitektur

tierten Klassen des Zeit- und Ressourcenkontexts wurden vorgestellt. Die Simulation greift auf diese Methoden zu, um die vorgestellten Algorithmen 4.1 und 4.2 umzusetzen. Die für die Evaluation produzierten Simulationsergebnisse wurden durch jeweils 250.000 Wiederholungen dieser Algorithmen erstellt. Die Simulationsdauer für einen einzelnen Prozess beträgt zwischen 6,2 und 8,5 Sekunden und zwischen 15,5 und 18,8 Sekunden für die gleichzeitige Simulation der Rechnungsein- und ausgangsprozesse[2]. Die Evaluation zeigt die Funktionsfähigkeit der entwickelten RTPN-Modelle, sowie deren Simulierbarkeit.

Zur Demonstration der Simulationsfähigkeiten der RTPN-Modelle wurden mehrere Prozesse vorgestellt, welche einzeln,und in Kombination simuliert wurden. Die Ergebnisse der Zeit- und Kostensimulation zeigen, dass Einflüsse zwischen Prozessen existieren. Im Vergleich zu der einzelnen Simulation jedes Prozesses zeigt sich, dass sich durch die Ressourcenabhängigkeiten die Ausführungszeit der Prozesse verändert. Der Nutzen der RTPN-Modelle besteht in der Simulation gleichzeitiger Prozesse, deren Ressourcenabhängigkeit und der Aufdeckung des Einflusses von Änderungen der Prozessstruktur und der Rechtezuweisung. Hiermit bieten RTPNs erstmals die Möglichkeit, Änderungen durch Sicherheitsanpassungen a priori,

[2]Verwendete Testumgebung: Ubuntu 12.04 – Kernel 3.11.0-26, Intel i7-3540M CPU mit 3.00GHz, 8 Gb Arbeitsspeicher, Java-Laufzeit Version 1.8.0_111-b14



(a) Zeitverhalten des unoptimierten Produktionsprozesses

(b) Zeitverhalten des optimierten Produktionsprozesses

Abbildung 5.22.: Unterschiedliche Zeitverhalten des Produktionsprozesses innerhalb der Prozessarchitektur

ganzheitlich auf das Zeitverhalten und somit auf die Verfügbarkeit der Prozesse zu untersuchen. Bisherige Verfahren unterstützten weder die gleichzeitige Simulation mehrere Prozesse, noch die Modellierung der Ressourcenabhängigkeiten oder der Rechtezuweisung.

Innerhalb der Beispiele wurde ein Prozess um eine zusätzliche Prüfaktivität ergänzt, um Betrugsfälle und Fehler durch Rechnungssteller zu vermeiden. Diese Prüfaktivität bindet Personal und erhöht den Zugriff auf geteilte Ressourcen. Eine isolierte Simulation des modifizierten Prozesses alleine zeigt, dass die Veränderung im Zeitablauf minimal sind. Die nachteiligen Auswirkungen dieser Änderung im Prozessablauf zeigen sich erst durch die simultane Simulation mit dem zweiten Prozess. Dieser zeigt durch die zusätzliche Personal- und Ressourcenbelastung eine eingeschränkte Abarbeitung, obwohl dieser Prozess unverändert blieb. Mit diesem Beispiel wurde durch die RTPNs simuliert, wie eine strukturelle Maßnahme, z.B. zur Wahrung der Gesetzeskonformität eines Prozesses, andere Prozesse beeinträchtigt.

Weiterhin wurden innerhalb der Evaluation Veränderungen der Prozessausführung durch ein geändertes Rechtemodell aufgedeckt. Aus dem letzten Beispiel wurden die Ausführungsrechte einer Aktivität umgestaltet. Für einen korrekten Prozessablauf wird nun gefordert, dass die Aktivität der Banküberweisung durch zwei Personen erfolgen muss. Dieses Vier-Augen Prinzip soll Betrugsfälle oder Fehler ver-

meiden, da sich die beteiligten Personen gegenseitig kontrollieren. Doch wegen der größeren Personalbindung führt auch diese Änderung zu Einflüssen in der Abarbeitung anderer Prozesse.

Um die Praxisnähe der Simulation zu zeigen, wurden die demonstrierten Auswirkungen auf die Prozesskosten berechnet. Das Ergebnis dieser Kostensimulation quantifiziert die Resultate der Zeitsimulation. Hiermit wird die in der Einleitung erwähnte Gefahr verdeutlicht, dass Änderungen eines Prozesses negative Auswirkungen auf andere Prozesse haben. Diese Auswirkungen können dazu führen, dass die durch eine Änderung erwünschten Vorteile wie erhöhte Sicherheit oder Konformität durch die negativen Auswirkungen konterkariert werden und daher verschiedene Interessensgruppen und Stakeholder den verstärkten Einsatz von sicheren Geschäftsprozessen ablehnen. Zur Beantwortung der Forschungsfragen ist es notwendig, solche Situationen zu erkennen. Durch den nun erstmals möglichen a priori Vergleich unterschiedlicher Variationen einer geforderten Sicherheitsumsetzungen werden Prozessentwickler befähigt, Lösungen bereitzustellen, welche akzeptable Auswirkungen hinsichtlich Zeit und Kosten zeigen, die Verfügbarkeit der Prozesse nicht gefährden und die Akzeptanz der Stakeholder finden.

Neben dem Aufdecken von Effekte durch Prozess-, Rechte- und Ressourcenänderungen, erlaubt die Simulation die Wahl von Abarbeitungsreihenfolgen, welche zu einer verbesserten Ressourcennutzung führen. Je nach Ablauffolge der Aktivitäten, sind Ressourcen zur Abarbeitung weiterer Aktivitäten blockiert oder frei. Eine Steuerung der Aktivitäten ermöglicht einen Zugriff auf die Ressourcen, sodass diese möglichst für weitere Aktivitäten frei sind. Je nach Gewichtung eines Prozesses kann diese Optimierung Ressourcen für die höher gewichteten Prozesse freihalten. Die Ergebnisse einer solchen Optimierung wurden an den Rechnungseingangs- und Rechnungsausgangsprozessen und auf eine Prozessarchitektur mit sieben Prozessen gezeigt. In beiden Fällen verzeichnet die Optimierung einen Erfolg. Die Ergebnisse decken auf, welche Aktivitätenreihenfolge die gegenseitige Beeinflussung von Prozessen minimal halten, um Ressourcen möglichst effizient nutzen. Hiermit wird die Verfügbarkeit der Prozesse gewährleistet, damit es zu keinem Unternehmensrisiko durch verzögerte Prozesse, damit verbundene Reputationsverluste oder verpassten Gelegenheiten kommt.

Kapitel 6

Zusammenfassung und Ausblick

Das Spannungsfeld der Unternehmen zwischen der Einhaltung von Sicherheit und Compliance und der Notwendigkeit, Geschäftsprozesse möglichst effizient zu gestalten, um die Interessen der Stakeholder zu wahren (Rappaport 1986), können Unternehmen nur durch detailliertes Wissen über das Zusammenwirken von Prozessen, Mitarbeiter, Zugriffskontrollen und Ressourcen lösen. Die durch Sicherheits- oder Compliancebestimmungen geforderten Anpassungen werden durch das Geschäftsprozessmanagement umgesetzt. Die Auswirkungen dieser Anpassungen limitieren sich nicht nur auf die geänderten Geschäftsprozesse, sondern auch auf abhängige Prozesse. Es kann nicht als gegeben gelten, dass den Prozessentwicklern diese Abhängigkeiten der Prozesse untereinander stets bekannt sind, weswegen die Auswirkungen im Vorhinein nicht gänzlich hinreichend sind. Im Rahmen dieser Arbeit wurde ein Verfahren zur Simulation dieser Auswirkungen a priori vorgestellt. Dies erlaubt Prozessentwicklern, verschiedene Umsetzung ihrer Sicherheitsmaßnahme miteinander zu vergleichen, um sowohl die entstehenden Kosten abschätzen zu können, als auch um die kosteneffizienteste Lösung zu finden.

Die bereits im Rahmen des Geschäftsprozessmanagements eingesetzte Möglichkeit, Auswirkungen von Änderungen der Geschäftsprozesse zu erkennen, ist die Simulation. Sie ermöglicht die Prognose des Verhaltens von Prozessen unter verschiedenen Szenarien und Prozessvariationen (Aalst 2015). Hierfür existiert eine Vielzahl unterschiedlicher Strategien zur Modellierung und Simulation von Geschäftsprozessen. Bestehende Verfahren nutzen jedoch Modelle, die nicht zur ganzheitlichen und gleichzeitigen Betrachtung der Rechteverwaltung, Ressourcennutzung und des

Zeitverhaltens von Geschäftsprozessen geeignet sind. Zusätzlich ermöglichen bisherige Verfahren nicht, nebenläufige Prozesse sowie gegenseitigen Abhängigkeiten zu simulieren. Genau diese Konzepte sind jedoch notwendig um die Auswirkungen oder Kosten von Prozessänderungen, hervorgerufen durch Sicherheits- oder Compliancegedanken, auf eine Prozessarchitektur zu erkennen und abzuschätzen. Die Auswirkungen von Prozessänderungen sind damit mit bestehenden Arbeiten nur lokal bestimmbar und nähern sich dementsprechend den real entstehenden Kosten nicht an. Nur eine Modellierung der Interdependenzen zwischen Prozessen erlaubt es, verschiedene Varianten einer Prozessanpassungen sinnvoll miteinander zu vergleichen, um möglichst effiziente Umsetzungen zu finden und den Aufwand einer geplanten Änderung zu quantifizieren.

Erst wenn Unternehmen in der Lage sind verschiedene Lösungen für die Umsetzung von Sicherheit oder Konformität zu betrachten, können diese geforderte Sicherheitsanpassungen möglichst effizient umsetzten. Gleichzeitig befähigen solche Lösungen, die Unternehmen dazu, zusätzliche Sicherheitsmaßnahmen einzuführen, die zwar sinnhaft sind, zuvor jedoch nicht umgesetzt wurden, da die dadurch entstehenden Kosten nicht abschätzbar waren oder nicht geklärt werden konnte, wie die Umsetzung effizient zu gestalten sei.

Da RTPN-Modelle die Modellierungsmöglichkeiten mehrerer Petri-Netz-Modelle kombinieren, steigt auch deren Komplexität. Es lässt sich kein Erreichbarkeitsgraph mehr bilden, mit dem es zum Beispiel möglich wäre, in Polynomialzeit zu berechnen, ob das Netz mit den gegebenen Ressourcen und Rechtevergaben ausführbar ist. Die Evaluation zeigt jedoch, dass die in einem RTPN enthaltenen Informationen nicht weiter abstrahierbar sind. Erst die Kombination aus Ressourcennutzung, zeitlicher Aspekte und der Möglichkeit simultane Prozesse abzubilden, ermöglicht die Änderungen von Rechtevergaben in der Prozessausführung zu simulieren.

6.1 Beitrag

Die im Rahmen dieser Arbeit entwickelten und vorgestellten Resource-Timed Petri Nets (RTPN) erfüllen die geforderte Kriterien der Simulation voneinander abhängiger Geschäftsprozesse. RTPNs kombinieren und erweitern die Fähigkeiten bestehender Petri-Netz-basierten Modelle zur Modellierung von Geschäftsprozessen.

Um die Simulation von Änderungen der Zugriffskontrolle und Rechteverwaltung zu ermöglichen, nutzen RTPN-Modelle einen Zugriffskontext, der eine rollenbasierte Rechtevergabe abbildet. Erstmals lassen sich Abhängigkeiten mehrerer Prozesse durch einen geteilten Zustand mit RTPN-Modellen beschreiben. Somit wird es ermöglicht die Auswirkungen von strukturellen Prozessänderungen, als auch von geänderten Zugriffskontrollen auf gegenseitig abhängige Geschäftsprozesse zu simulieren.

Hierdurch wird es ermöglicht, die gesamten Auswirkungen durch eine Prozessänderungen nachzuvollziehen. Mit den aus der Simulation gewonnenen Informationen lassen sich die Kosten einer Prozessänderung auf die gesamte Prozessarchitektur abschätzen und somit eine Abwägung über die Sinnhaftigkeit dieser Änderung treffen. Die Simulation eines RTPN-Modells erlaubt hiermit den Vergleich unterschiedlicher Sicherheitsumsetzungen, um eine möglichst kosteneffiziente Variante zu finden.

Die Aussagekraft der Simulation von Geschäftsprozessen durch ein RTPN-Modell steigt mit der Genauigkeit der modellierten Ressourcennutzung und des Zeitverhaltens der Aktivitäten. Während die vorherrschende Zugriffskontrolle des Unternehmen bekannt ist, sind es die Ressourcen und der Zeitbedarf einzelner Aktivitäten nicht zwingend. Innerhalb dieser Dissertation wurden Verfahren vorgestellt, diese Informationen aus Prozess-Logs zu extrahieren. Da Prozess-Logs die reale Abarbeitung der Aktivitäten widerspiegeln, sind auch die hieraus gewonnen Informationen realitätsnah, um genaue Prognosen durch die Simulation durch ein RTPN-Modell zu ermöglichen.

Die Simulation von Geschäftsprozessen durch RTPNs ist nicht darauf begrenzt, Prognosen zu generieren. Die in der Simulation evaluierten Abarbeitungsreihenfolgen lassen sich für eine Optimierung der Prozessabläufe nutzen. Die Evaluation zeigt, dass eine vorhandene Prozessarchitektur aufgrund unterschiedlicher Abarbeitungsreihenfolgen unterschiedliche Verhalten aufweist. Eine vorgestellte Optimierungsstrategie wählt Abarbeitungsreihenfolgen, die zu einer Ressourcenauslastung mit geringen Wechselwirkungen mit anderen Prozessen führt. Mit diesen Ergebnissen lassen sich Prozessarchitekturen steuern, um Sicherheitsanpassungen oder Prozessänderungen im Allgemeinen effizient in eine bestehende Prozessarchitektur zu integrieren.

Die RTP-Netze bilden eine Grundlage für weitere Forschungsarbeiten. So wäre es möglich, die Wahrscheinlichkeit zu berechnen, dass ein Prozess Störungen erfährt. Hierfür ließen sich Informationen der abgebrochenen Aktivitäten aus einem Prozess-Log Extrahieren. Die Simulation könnte dann für gegebene Prozesse sowie deren Ablaufhistorie die Wahrscheinlichkeit für einen Abbruch beziffern. Auch die Optimierung von Geschäftsprozessen lässt sich weiter verbessern, zum Beispiel durch den Einsatz genetischer Algorithmen, welche die Abarbeitungsreihenfolge umsortieren. Die offene Plattform SWAT, in welcher die RTPNs implementiert wurde, erlaubt eine solche Erweiterung durch den modularen Aufbau.

6.2 Anwendungsfälle

Mögliche Einsatzszenarien der entwickelten RTPN-Modelle sind (teil-)automatisierte Prozessarchitekturen, welche durch eine zentrale Instanz, wie ein WfMS, gesteuert werden. In solch einem Umfeld sind die Prozesse und deren Variationen genügend spezifiziert, um einerseits die Simulation zu ermöglichen und andererseits existieren Protokolle der Prozessabläufe, aus denen die erforderlichen Daten zur Gestaltung der RTPN-Modelle extrahierbar sind. Landesweite Projekte wie GESINE zeigen die Einsatzfähigkeit solcher automatisierten Prozesse. Die industrielle Wandlung durch die geforderte Automatisierung und Standardisierung durch Industrie 4.0 begünstigt in Zukunft die Einsatzfähigkeit der gezeigten Simulationstechnik.

In solchen Szenarien erlauben die RTPN-Modelle einem Unternehmen, das Spannungsfeld zwischen Sicherheit, Compliance und Verfügbarkeit aufzulösen und Sicherheitsmaßnahmen durchzuführen, welche zuvor aus Kostengründen von unterschiedlichen Interessensgruppen hätten abgelehnt werden können.Die Umgestaltung von Prozessen zum Erreichen von Konformität oder Erhöhung der Sicherheit lässt sich nun so verwirklichen, dass die Auswirkungen auf die Prozessarchitektur möglichst minimal sind. Die RTPN-Modelle stellen durch ihre Simulationsfähigkeit somit ein Werkzeug für Unternehmen dar, um die Bereitschaft zur Stärkung der Sicherheit zu erhöhen.

Anhang A

Veröffentlichungen

Im Rahmen dieser Dissertation und weiteren Projekttätigkeiten entstanden folgende Veröffentlichungen:

Accorsi, Rafael, Julius Holderer, Thomas Stocker und Richard M. Zahoransky (2014). "Security Workflow Analysis Toolkit". In: *Sicherheit*. Hrsg. von Stefan Katzenbeisser, Volkmar Lotz und Edgar R. Weippl. Bd. 228. LNI. GI, S. 433–442. ISBN: 978-3-88579-622-0.

Eymann, Torsten, Martin Jurisch, Günter Müller, Dennis Schmidt, Philipp Vogler und Richard M. Zahoransky (2014). "Die etwas andere Spezialeinheit: SWAT (Security Workflow Analysis Toolkit) zur Sicherung von Geschäftsprozessen". In: *Wissenschaft Trifft Praxis*, S. 26.

Rechert, K., K. Meier, R.M. Zahoransky, D. Wehrle, D. von Suchodoletz, B. Greschbach, S. Wohlgemuth und I. Echizen (2013). "Reclaiming Location Privacy in Mobile Telephony Networks;Effects and Consequences for Providers and Subscribers". In: *IEEE Systems Journal* 7.2, S. 211–222. ISSN: 1932-8184. DOI: 10.1109/JSYST.2013.2241357.

Zahoransky, Richard M., Klaus Rechert, Konrad Meier, Dennis Wehrle und Dirk Von Suchodoletz (2012). "Cellular Location Determination-Reliability and Trustworthiness of GSM Location Data." In: *ARCS Workshops*. Bd. 200. Citeseer, S. 63–73.

Zahoransky, Richard M., Saher Semaan und Klaus Rechert (2013). "Identity and Access Management for Complex Research Data Workflows." In: *DFN-Forum Kommunikationstechnologien*, S. 107–117.

Zahoransky, Richard M. und Saher Semaan (2014a). "bwIDM: Anbindung nicht-webbasierter IT-Infrastrukturen an eine SAML/Shibboleth-Föderation". In: *Hochleistungsrechnen in Baden-Wuerttemberg-Ausgewaehlte Aktivitaeten im bwGRiD 2012: Beitraege zu Anwenderprojekten und Infrastruktur im bwGRiD im Jahr 2012*, S. 161.

Zahoransky, Richard M., Thomas Koslowski und Rafael Accorsi (2014b). "Toward Resilience Assessment in Business Process Architectures". In: *Computer Safety, Reliability, and Security: SAFECOMP 2014 Workshops: ASCoMS, DECSoS, DEVVARTS, ISSE, ReSA4CI, SASSUR. Florence, Italy, September 8-9, 2014. Proceedings*. Hrsg. von Andrea Bondavalli, Andrea Ceccarelli und Frank Ortmeier. Cham: Springer International Publishing, S. 360–370. ISBN: 978-3-319-10557-4. DOI: 10.1007/978-3-319-10557-4_39. URL: http://dx.doi.org/10.1007/978-3-319-10557-4_39.

Zahoransky, Richard M., Christian Brenig und Thomas Koslowski (Aug. 2015). "Towards a Process-Centered Resilience Framework". In: *2015 10th International Conference on Availability, Reliability and Security*, S. 266–273. DOI: 10.1109/ARES.2015.68.

Zahoransky, Richard M., Julius Holderer, Adrian Lange und Christian Brenig (2016). "Process Analysis as First Step Towards Automated Business Security". In: *European Conference on Information Systems* 24.

Anhang B

Ressourcenübersicht der Beispielprozesse

Dieser Abschnitt enthält Beschreibungen und Ressourcennutzung der in der Evaluation verwendeten Prozesse.

Bestell- und Produktionsprozess

Der Auftragseingang in Abb. 5.4a stellt den größten Prozess innerhalb der Architektur dar. Dieser Prozess wird gestartet, wenn ein Kunde eine Bestellung aufgegeben hat. Daraufhin wird die bestellte Ware produziert und nach einer Abschließenden Qualitätskontrolle an den Kunden ausgeliefert. Wenn die für ein Produkt erforderlichen Teile bereits vormontiert sind, müssen diese nur noch zusammengefügt werden. Ansonsten wird das Produkt aus den Einzelteilen montiert. Dabei kann es vorkommen, dass nicht vorhandene Einzelteile bestellt werden müssen. Eintreffende Einzelteile werden ebenfalls einer Qualitätskontrolle unterworfen. Die einzelnen Aktivitäten und deren Ressourcennutzung sind in Tabelle B.1 aufgezeigt.

Rechnungsausgang

Abbildung 5.4b stellt einen Rechnungsprozess dar, der so oder nur mit leichten Änderungen in vielen Unternehmen ablaufen könnte. Dabei wird geprüft, auf welche Art eine Rechnung an einen Kunden versendet werden soll. Diese wird dann per E-Mail oder postalisch an den Kunden versendet. In einer Datenbank wird dies

Aktivität	Rollen	Ressource
Auftragseingang	Sekretariat, Poststelle	Post
Baugruppenplan drucken	Fertigungsabteilung, Kontruktionsabteilung	Drucker ∧ DB
Montageplan erstellen	Fertigung	Drucker ∧ DB
Stückliste	Fertigung	Drucker ∧ DB ∧ Netzlaufwerk
Teile beschaffen	Fertigung, Montageabteilung	-
Bestellung vorbereiten	Fertigung	(Drucker, DB) ∨ (E-Mail, DB)
Bestellung aufgeben	Ein- und Verkauf	((Drucker ∧ Postausgang) ∨ E-Mail) ∧ DB
Inspektion	Qualitätsmanagement	Post ∧ Netzlaufwerk ∧ DB ∧ Drucker ∧ Montageplatz
Lagerbestand prüfen	Montage, Fertigung, Lager, Einkauf	DB
aus Lager entnehmen	Lager	Lager ∧ DB
Teile montieren	Montage	Montagewerkzeug ∧ Montageplatz
Baugruppen beschaffen	Fertigung	Lager ∧ DB
Baugruppen montieren	Fertigung	Montagewerkzeug ∧ Montageplatz
Finale Qualitätskontrolle	Qualitätsmanagement	Post ∧ Netzlaufwerk ∧ DB ∧ Drucker ∧ Montageplatz
Versand und Rechnungsstellung	Ein- und Verkauf	DB ∧ Post ∧ Drucker ∧ E-Mail

Tabelle B.1.: Ressourcennutzung der Aktivitäten des Bestell- und Produktionsprozesses

vermerkt. Die Ressourcen, Betriebsmittel und Zugangsnutzung sind in Tabelle B.2 hinterlegt.

Rechnungseingang

In Abbildung 5.4c wird ein Rechnungseingang gezeigt. Die Rechnung trifft elektronisch oder postalisch ein. Der Eingang wird in der Datenbank protokolliert und der Betrag überwiesen. Der Datenbankeintrag der Rechnung wird daraufhin als erledigt markiert. Tabelle B.3 zeigt die Aktivitäten und Ressourcennutzung des Prozesses.

Aktivität	Subjekt	Ressource
Kundenprofil öffnen	Ein- und Verkauf, Sekretariat	DB
Kundenpräferenz beachten	Ein- und Verkauf, Sekretariat	DB
E-Mail versenden	alle	E-Mail
Brief versenden	Sekretariat	Post ∧ Drucker
Scannen	alle	Netzlaufwerk, Multifunktionsgerät
Kundenprofil aktualisieren	Ein- und Verkauf, Sekretariat	DB

Tabelle B.2.: Ressourcennutzung der Aktivitäten des Rechnungsausgangsprozesses

Aktivität	Subjekt	Ressource
E-Mail öffnen	Ein- und Verkauf, Sekretariat	E-Mail
Post öffnen	Ein- und Verkauf, Sekretariat	Post
Rechnung hinzufügen	Sekretariat	DB
Banking	Ein- und Verkauf, Sekretariat	DB ∧ Drucker
Bezahlung hinterlegen	Ein- und Verkauf, Sekretariat	DB
Rechnung prüfen	Ein- und Verkauf ∧ Sekretariat	DB ∧ Drucker

Tabelle B.3.: Ressourcennutzung des Rechnungseingangsprozesses

Warenbestellung

Der Prozess in Abbildung 5.4d stammt aus Log-Daten einer süd-deutschen Firma (Zahoransky et al. 2016). Vor einer Bestellung muss eine Bedarfsanforderung erfolgen. Diese wird freigegeben oder muss geändert werden. Danach werden die Bestellungen aufgegeben. Es wird davon ausgegangen, dass die Ware nach dem Eingang der Rechnung eintrifft.

Regelmäßige Sicherung

In Abbildung 5.4e ist ein Beispiel für eine regemäßige Sicherung der Datenbank beschrieben. Diese wird automatisch in einem bestimmten Intervall gestartet. Da-

nach wird die Datenbank auf Konsistenz geprüft, ausgelesen und an einem anderen Speicherort abgelegt. Das Resultat der Datensicherung wird per E-Mail versandt. Die Aktivitäten und verwendeten Ressourcen des Prozesses sind in Tabelle B.4 aufgelistet.

Aktivität	Subjekt	Ressource
Integritätsprüfung	automatisiert	`DB`
SQL Dump	automatisiert	`DB`
Backup kopieren	automatisiert	`Netzlaufwerk`
Backup bestätigen	Technik	`E-Mail` $\wedge$ `Netzlaufwerk`

Tabelle B.4.: Ressourcennutzung der regelmäßigen Stammdatensicherung

Wartungsarbeiten

Als Beispiel für regelmäßige Wartungsarbeiten wurde der Prozess aus Abbildung 5.4f zur Architektur hinzugefügt. In diesem Prozess wird der Abteilungsdrucker gewartet oder eine Störung behoben. Die Aktivitäten und die zugehörigen Ressourcen sind in Tabelle B.5 verzeichnet.

Aktivität	Subjekt	Ressource
Drucker öffnen	Technik	`Drucker`
Papier aufstocken	Technik	`Lager`$\wedge$ `Drucker`
Toner wechseln	Technik	`Lager`, `Drucker`, `Werkzeug`
Papierstau beheben	Technik	`Werkzeug`
Testen	Technik	`Drucker`

Tabelle B.5.: Ressourcennutzung der Wartungsarbeiten am Drucker

Kompensierung von Ausfällen

Bei einem defektem Teil oder defektem Produkt wendet sich der Kunde an das Unternehmen und fordert Kompensation. Dabei wird ein Kompensationswunsch geprüft. Darauf hin wird entschieden, ob das Produkt ausgetauscht oder repariert wird. Das reparierte oder ausgetauschte Produkt wird versandt, worauf hin der Prozess endet. Tabelle B.6 zeigt die Aktivitätenabfolge und Ressourcennutzung.

Aktivität	Subjekt	Ressource
Prüfe erhaltenes Produkt	Qualitätsmanagement	Post, DB, Drucker
Kostenschätzung	Fertigung, Montage	DB, E-Mail
Reparaturangebot senden	Sekretariat	(Post, DB), (E-Mail, DB)
Kompensation	Ein- und Verkauf	Lager, DB, Post
Reparatur	Fertigung, Montage	Montagewerkzeug, Montageplatz, Drucker
Dokumentation der Reparatur	Fertigung, Montage	DB, Drucker, E-Mail
Versand	Ein- und Verkauf	DB, Post, Drucker, E-Mail

Tabelle B.6.: Ressourcennutzung bei Kompensierung

Ressourcenüberblick

Die innerhalb der Architektur verwendeten Ressourcen und Betriebsmittel sowie Personal sind in Tabelle B.7 zusammengefasst.

Ressource	Art	Verfügbarkeit
Poststelle	normal	einfach
Drucker	mehrfach (2 mal)	Multifunktionsgerät, Abteilungsdrucker
DB	geteilt	5 gleichzeitige Zugriffe
Netzlaufwerk	geteilt	3 gleichzeitige Zugriffe
E-Mail	geteilt	10 gleichzeitige Zugriffe
Montageplatz	mehrfach	Montageplatz 1...3
Lager	normal	einfach
Montagewerkzeug	mehrfach	Montagewerkzeug 1...2
Multifunktionsgerät	normal	einfach
Sekretariat	Rolle	Anne, Bart
Poststelle	Rolle	Bart
Konstruktion	Rolle	Claus, Daniel
Ein- und Verkauf	Rolle	Eve
Qualitätsmanagement	Rolle	Frank, Gerlinde
Lagerabteilung	Rolle	Claus

Tabelle B.7.: Übersicht über die verwendeten Ressourcen und Rollenteilnehmer

Anhang C

Zeitkontext der Beispielprozesse

In diesem Abschnitt werden die angenommenen Zeitdauern der Aktivitäten aufgelistet.

C.1 Produktionsprozess

Für die Aktivitäten des Produktionsprozesses werden die in Tabelle C.1 aufgelisteten Zeitverhalten angenommen.

C.2 Rechnungsausgang

In Tabelle C.2 sind die angenommen Zeitdauer für die Aktivitäten des Rechnungsausgangsprozesses hinterlegt.

C.3 Rechnungseingang

Die Aktivitäten des Rechnungseingangs zeigen die in Tabelle C.3 definierten Zeitverhalten.

Aktivität	Zeitverhalten
Auftragseingang	$ln\mathcal{N}(3,0,3)$
Baugruppenplan drucken	$\mathcal{N}(7,1)$
Montageplan erstellen	$\mathcal{N}(6,1)$
Stückliste erstellen	$\Gamma(7,0,7)$
Teile beschaffen	$\Gamma(8,0,8)$
Bestellung vorbereiten	$ln\mathcal{N}(2,0,5)$
Bestellung aufgeben	$[720-1440]$
Inspektion	$ln\mathcal{N}(3,0,2)$
Lagerbestand prüfen	$ln\mathcal{N}(2,0,4)$
aus Lager entnehmen	$\Gamma(3,1)$
Teile montieren	$ln\mathcal{N}(4,0,2)$
Baugruppen beschaffen	$ln\mathcal{N}(4,0,2)$
Baugruppen montieren	$ln\mathcal{N}(4,0,4)$
Finale Qualitätskontrolle	$\Gamma(10,1)$
Versand und Rechnungsstellung	$\Gamma(10,5)$

Tabelle C.1.: Zeitverhalten der Aktivitäten des Bestell- und Produktionsprozesses

Aktivität	Zeitverhalten
Kundenprofil öffnen	$\mathcal{N}(2,0,25)$
Kundenpräferenz beachten	$ln\mathcal{N}(0,25,0,1)$
E-Mail versenden	$\mathcal{N}(25,5)$
Brief versenden	$\mathcal{N}(32,5)$
Scannen	$\Gamma(3,3)$
Kundenprofil aktualisieren	$ln\mathcal{N}(3,0,5)$

Tabelle C.2.: Zeitverhalten der Aktivitäten des Rechnungsausgangsprozesses

Aktivität	Zeitverhalten
E-Mail öffnen	$\mathcal{N}(1,1)$
Post öffnen	$\mathcal{N}(30,5)$
Scannen	$\Gamma(3,3)$
Rechnung hinzufügen	$\mathrm{Exp}(5)$
Banking	$\mathcal{N}(12,4)$
Bezahlung hinterlegen	$\mathcal{N}(3,1)$
Rechnung prüfen	$ln\mathcal{N}(2,5,0,2)$

Tabelle C.3.: Zeitverhalten des Rechnungseingangsprozesses

C.4 Warenbestellung

Für die Aktivitäten der Warenbestellung werden die in Tabelle C.4 aufgelisteten Zeitverhalten angenommen.

Aktivität	Zeitverhalten
Bestellanforderung	$ln\mathcal{N}(2, 0, 1)$
Bestellung	$\Gamma(3, 1)$
Forderung freigeben	$\mathcal{N}(5, 1)$
Lieferung	$[720 - 1440]$
Rechnungseingang	$\mathcal{N}(10, 2)$
Wareneingang	$\mathcal{N}(60, 10)$

Tabelle C.4.: Zeitverhalten der Aktivitäten der Warenbestellung

C.5 Regelmäßige Sicherung

Die Aktivitäten zur Kopie der Kundendaten zeigen die in Tabelle C.5 definierten Zeitverhalten.

Aktivität	Zeitverhalten
Integritätsprüfung	$[6 - 8]$
SQL Dump	$ln\mathcal{N}(2, 0, 1)$
Backup kopieren	$ln\mathcal{N}(1, 5, 0, 25)$
Backup bestätigen	$\Gamma(3, 1)$

Tabelle C.5.: Zeitverhalten der regelmäßigen Stammdatensicherung

C.6 Wartungsarbeiten

Die Aktivitäten der Wartungsarbeiten zeigen die in Tabelle C.6 definierten Zeitverhalten.

Aktivität	Zeitverhalten
Drucker öffnen	$\mathcal{N}(5, 2)$
Papier aufstocken	$\Gamma(5, 1)$
Toner wechseln	$ln\mathcal{N}(2, 5, 0, 25)$
Papierstau beheben	Weibull(3, 2)
Testen	Weibull(3,2)

Tabelle C.6.: Zeitverhalten der Wartungsarbeiten am Drucker

C.7 Kompensierung von Ausfällen

Die Aktivitäten des Kompensationsprozesses zeigen die in Tabelle C.7 definierten Zeitverhalten.

Aktivität	Zeitverhalten
Prüfe erhaltenes Produkt	$\Gamma(20, 3)$
Kostenschätzung	$ln\mathcal{N}(2, 5, 0, 2)$
Reparaturangebot senden	—
Kompensation	$\mathcal{N}(6, 1, 5)$
Reparatur	$\Gamma(60, 1, 5)$
Dokumentation der Reparatur	$\mathcal{N}(10, 2)$
Versand	$ln\mathcal{N}(2, 5, 0, 1)$

Tabelle C.7.: Ressourcennutzung bei Kompensierung

Literatur

Aalst, Wil M.P. van der (1993). *Interval timed coloured Petri nets and their analysis*. Springer.

— (1996). "Three Good reasons for Using a Petri-net-based Workflow Management System". In: *Proceedings of the International Working Conference on Information and Process Integration in Enterprises (IPIC'96)*. Hrsg. von S. Navathe und T. Wakayama, S. 179–201.

— (1997). "Verification of workflow nets". In: *International Conference on Application and Theory of Petri Nets*. Springer, S. 407–426.

— (1998a). "The application of Petri nets to workflow management". In: *Journal of circuits, systems, and computers* 8.01, S. 21–66.

— (1998b). "Three good reasons for using a Petri-net-based workflow management system". In: *Information and Process Integration in Enterprises*. Springer, S. 161–182.

— (2010). "Business Process Simulation Revisited". In: *Enterprise and Organizational Modeling and Simulation: 6th International Workshop, EOMAS 2010, held at CAiSE 2010, Hammamet, Tunisia, June 7-8, 2010. Selected Papers*. Hrsg. von Joseph Barjis. Berlin, Heidelberg: Springer Berlin Heidelberg, S. 1–14. ISBN: 978-3-642-15723-3. DOI: 10.1007/978-3-642-15723-3_1. URL: http://dx.doi.org/10.1007/978-3-642-15723-3_1.

— (2011). *Process mining: discovery, conformance and enhancement of business processes*. Springer Science & Business Media. Kap. 4.

— (Aug. 2012). "Process Mining". In: *Commun. ACM* 55.8, S. 76–83. ISSN: 0001-0782. DOI: 10.1145/2240236.2240257. URL: http://doi.acm.org/10.1145/2240236.2240257.

— (2015). "Business Process Simulation Survival Guide". In: *Handbook on Business Process Management 1: Introduction, Methods, and Information Systems*. Hrsg. von Jan vom Brocke und Michael Rosemann. Berlin, Heidelberg: Springer Berlin Heidelberg, S. 337–370. ISBN: 978-3-642-45100-3. DOI: 10.1007/978-3-642-45100-3_15. URL: http://dx.doi.org/10.1007/978-3-642-45100-3_15.

Aalst, Wil M.P. van der (2016). *Process Mining: Data Science in Action*. Springer.

Aalst, Wil M.P. van der und Michiel A. Odijk (1995). "Analysis of railway stations by means of interval timed coloured Petri nets". In: *Real-time systems* 9.3, S. 241–263.

Aalst, Wil M.P. van der, Arthur H. M. Hofstede und Mathias Weske (2003). "Business Process Management: A Survey". In: *Business Process Management: International Conference, BPM 2003 Eindhoven, The Netherlands, June 26–27, 2003 Proceedings*. Hrsg. von Wil M.P. van der Aalst und Mathias Weske. Berlin, Heidelberg: Springer Berlin Heidelberg, S. 1–12. ISBN: 978-3-540-44895-2. DOI: 10.1007/3-540-44895-0_1. URL: http://dx.doi.org/10.1007/3-540-44895-0_1.

Aalst, Wil M.P. van der und Kees Max Van Hee (2004). *Workflow management: models, methods, and systems*. MIT press.

Aalst, Wil M.P. van der und Arthur H. M. ter Hofstede (2005). "YAWL: yet another workflow language". In: *Information systems* 30.4, S. 245–275.

Aalst, Wil M.P. van der, Arya Adriansyah, Ana Karla Alves de Medeiros, Franco Arcieri, Thomas Baier, Tobias Blickle, Jagadeesh Chandra Bose, Peter van den Brand, Ronald Brandtjen, Joos Buijs et al. (2011). "Process mining manifesto". In: *Business process management workshops*. Springer, S. 169–194.

Accorsi, Rafael (2013). "Security in Business Process Management". In: *it–Information Technology it–Information Technology* 55.6, S. 215–216.

Accorsi, Rafael und Claus Wonnemann (2011a). "Forensic leak detection for business process models". In: *Advances in Digital Forensics VII*. Springer, S. 101–113.

— (2011b). "InDico: Information flow analysis of business processes for confidentiality requirements". In: *Security and Trust Management*. Berlin, Heidelberg: Springer, S. 194–209.

Accorsi, Rafael, Meike Ullrich und Wil M.P. van der Aalst (2012). "Process mining". In: *Informatik-Spektrum* 35.5, S. 354–359.

Accorsi, Rafael, Julius Holderer, Thomas Stocker und Richard M. Zahoransky (2014). "Security Workflow Analysis Toolkit". In: *Sicherheit*. Hrsg. von Stefan Katzenbeisser, Volkmar Lotz und Edgar R. Weippl. Bd. 228. LNI. GI, S. 433–442. ISBN: 978-3-88579-622-0.

Alberti, Marco, Federico Chesani, Marco Gavanelli, Evelina Lamma, Paola Mello und Paolo Torroni (Aug. 2008). "Verifiable Agent Interaction in Abductive Logic Programming: The SCIFF Framework". In: *ACM Trans. Comput. Logic* 9.4, 29:1–29:43. ISSN: 1529-3785. DOI: 10.1145/1380572.1380578. URL: http://doi.acm.org/10.1145/1380572.1380578.

Alisch, Katrin, Eggert Winter und Ute Arentzen (2013). *Gabler Wirtschafts Lexikon*. Springer-Verlag.

Allweyer, Thomas (2015). *BPMN 2.0-Business Process Model and Notation: Einführung in den Standard für die Geschäftsprozessmodellierung*. BoD–Books on Demand.

April, Jay, Marco Better, Fred Glover, James Kelly und Manuel Laguna (2006). "Enhancing business process management with simulation optimization". In: *Proceedings of the 38th conference on Winter simulation*. Winter Simulation Conference, S. 642–649.

AWV e.V. (2014). *Das ZUGFeRD-Format, Spezifikation und Umsetzungsregeln zum branchen- übergreifenden Kern-Rechnungsformat des Forums elektronische Rechnung Deutschland (FeRD)*.

Baier, C., B. Haverkort, H. Hermanns und J. P. Katoen (2003). "Model-checking algorithms for continuous-time Markov chains". In: *IEEE Transactions on Software Engineering* 29.6, S. 524–541. ISSN: 0098-5589. DOI: 10.1109/TSE.2003.1205180.

Baltzer, Björn (2007). "Time-driven Activity-based Costing: Entwicklung, Methodik, Anwendungsfelder". In:

Barizo, Dee (2014). *The 10 Worst Accounting Scandals of All Time*. URL: http://www.accounting-degree.org/scandals/ (besucht am 24. 01. 2017).

Barnett, Mike (2003). "Modeling & simulation in business process management". In: *BP Trends Newsletter, White Papers & Technical Briefs* 10.1.

Becker, Jörg, Christoph Mathas und Axel Winkelmann (2009). *Geschäftsprozessmanagement*. Springer-Verlag.

Becker, Jörg, Martin Kugeler und Michael Rosemann (2013). *Process management: a guide for the design of business processes*. Springer Science & Business Media.

Berthomieu, Bernard und Miguel Menasche (1983). "An Enumerative Approach For Analyzing Time Petri Nets". In: *Proceedings IFIP*. Elsevier Science Publishers, S. 41–46.

Best, Eva und Martin Weth (2010). *Process Excellence*. Gabler Verlag / Springer Fachmedien Wiesbaden GmbH, Wiesbaden. ISBN: 978-3-8349-221-3.

Biffar, Jürgen (2016). *Auf dem Weg zum digitalen Büro: Bitkom Digital Office Index 2016*. URL: \url{https://www.bitkom.org/Presse/Presseinformation/Der-Weg-zum-digitalen-Buero-ist-erst-zur-Haelfte-geschafft.html} (besucht am 08. 03. 2017).

Bishop, Matt (2004). *Introduction to Computer Security*. Addison-Wesley Professional. ISBN: 0321247442.

Bishop, Matthew A (2002). *The art and science of computer security*. Addison-Wesley Longman Publishing Co., Inc.

BMWi, Bundesministerium für Wirtschaft und Energie (2014). *Faktenblatt Förderinitiative eStandards*.

Botha, R. A. und J. H. P. Eloff (2001). "Separation of duties for access control enforcement in workflow environments". In: *IBM Systems Journal* 40.3, S. 666–682. ISSN: 0018-8670. DOI: 10.1147/sj.403.0666.

Brown, Scott W. (1997). "Attentional resources in timing: Interference effects in concurrent temporal and nontemporal working memory tasks". In: *Perception & Psychophysics* 59.7, S. 1118–1140. ISSN: 1532-5962. DOI: 10.3758/BF03205526. URL: http://dx.doi.org/10.3758/BF03205526.

Browning, Tyson R. (2003). "On customer value and improvement in product development processes". In: *Systems Engineering* 6.1, S. 49–61. ISSN: 1520-6858. DOI: 10.1002/sys.10034. URL: http://dx.doi.org/10.1002/sys.10034.

Browning, Tyson R. und S. D. Eppinger (2002). "Modeling impacts of process architecture on cost and schedule risk in product development". In: *IEEE Transactions on Engineering Management* 49.4, S. 428–442. ISSN: 0018-9391. DOI: 10.1109/TEM.2002.806709.

Brunner, Franz J (2014). *Japanische Erfolgskonzepte: KAIZEN, KVP, Lean Production Management, Total Productive Maintenance Shopfloor Management, Toyota Production Management, GD3-Lean Development*. Carl Hanser Verlag GmbH Co KG.

Bundeskriminalamt (2015). *Bundeslagebild Wirtschaftskriminalität 2015*.

Chiola, G., C. Dutheillet, G. Franceschinis und S. Haddad (1991). "On Well-Formed Coloured Nets and Their Symbolic Reachability Graph". In: *High-level Petri Nets: Theory and Application*. Hrsg. von Kurt Jensen und Grzegorz Rozenberg. Berlin, Heidelberg: Springer Berlin Heidelberg, S. 373–396. ISBN: 978-3-642-84524-6. DOI: 10.1007/978-3-642-84524-6_13. URL: http://dx.doi.org/10.1007/978-3-642-84524-6_13.

Claussen, Jens (2011). *Compliance- Oder Integrity-Management: Massnahmen Gegen Korruption in Unternehmen*. Metropolis.

Cook, Jonathan E und Alexander L Wolf (1999). "Software process validation: quantitatively measuring the correspondence of a process to a model". In: *ACM Transactions on Software Engineering and Methodology (TOSEM)* 8.2, S. 147–176.

Cooper, Robin (1988). "The Rise of Activity-Based Costing - Part one: What Is An Activity-Based Cost System". In: *Journal of Cost Management*.

Cooper, Robin und Robert S Kaplan (1988). "Measure costs right: make the right decisions". In: *Harvard business review* 66.5, S. 96–103.

Denning, Dorothy E und Peter J Denning (1977). "Certification of programs for secure information flow". In: *Communications of the ACM* 20.7, S. 504–513.

Devroye, Luc (1986). "Sample-based non-uniform random variate generation". In: *Proceedings of the 18th conference on Winter simulation*. ACM, S. 260–265.

Dickmann, Philipp (2008). *Schlanker Materialfluss: mit Lean Production, Kanban und Innovationen*. Springer-Verlag.

Diederichs, Marc (2013). *Risikomanagement und Risikocontrolling*. Vahlen.

Dijkman, Remco M., Marlon Dumas und Chun Ouyang (2008). "Semantics and analysis of business process models in {BPMN}". In: *Information and Software Technology* 50.12, S. 1281 –1294. ISSN: 0950-5849. DOI: http://dx.doi.org/10.1016/j.infsof.2008.02.006. URL: http://www.sciencedirect.com/science/article/pii/S0950584908000323.

Dombrowski, Uwe und Tim Mielke (2015). *Ganzheitliche Produktionssysteme: Aktueller Stand und zukünftige Entwicklungen*. Springer-Verlag.

Dongen, Boudewijn F van und Wil M.P. van der Aalst (2005). "A Meta Model for Process Mining Data." In: *EMOI-INTEROP* 160, S. 30.

Dongen, Boudewijn F. van und Wil M.P. van der Aalst (2005). "A Meta Model for Process Mining Data". In: *EMOI-INTEROP*. Hrsg. von Michele Missikoff und Antonio De Nicola. Bd. 160. CEUR Workshop Proceedings. CEUR-WS.org.

Dumas, Marlon und Arthur H. M. ter Hofstede (2001). "UML Activity Diagrams as a Workflow Specification Language". In: *UML 2001 — The Unified Modeling Language. Modeling Languages, Concepts, and Tools: 4th International Conference Toronto, Canada, October 1–5, 2001 Proceedings*. Hrsg. von Martin Gogolla und Cris Kobryn. Berlin, Heidelberg: Springer Berlin Heidelberg, S. 76–90. ISBN: 978-3-540-45441-0. DOI: 10.1007/3-540-45441-1_7. URL: http://dx.doi.org/10.1007/3-540-45441-1_7.

Elter, Constanze (2014). "Elektronische Rechnungen". In: *Rechnung stellen - Umsatz sichern: Alle Vorschriften mit Tipps und Beispielen*. Wiesbaden: Springer Fachmedien Wiesbaden, S. 35–43. ISBN: 978-3-658-03217-3. DOI: 10.1007/978-3-658-03217-3_3. URL: http://dx.doi.org/10.1007/978-3-658-03217-3_3.

Eymann, Torsten, Martin Jurisch, Günter Müller, Dennis Schmidt, Philipp Vogler und Richard M. Zahoransky (2014). "Die etwas andere Spezialeinheit: SWAT (Security Workflow Analysis Toolkit) zur Sicherung von Geschäftsprozessen". In: *Wissenschaft Trifft Praxis*, S. 26.

Farnsworth, David L (2013). "THE MEMORYLESS PROPERTY". In: *Mathematics and Computer Education* 47.2, S. 99.

Feller, Willliam (2008). *An introduction to probability theory and its applications*. Bd. 2. John Wiley & Sons.

Feltham, Brett, Mandi Xu et al. (2013). "Lessons for managing restructures and avoiding'sham'redundancies". In: *Keeping Good Companies* 65.10, S. 620.

Ferraiolo, David, D Richard Kuhn und Ramaswamy Chandramouli (2003). *Role-based access control*. Artech House.

Ferraiolo, David F., Ravi Sandhu, Serban Gavrila, D. Richard Kuhn und Ramaswamy Chandramouli (Aug. 2001). "Proposed NIST Standard for Role-based Access Control". In: *ACM Trans. Inf. Syst. Secur.* 4.3, S. 224–274. ISSN: 1094-9224. DOI: 10.1145/501978.501980. URL: http://doi.acm.org/10.1145/501978.501980.

Finkeißen, Alexander (1999). *Prozess-Wertschöpfung: Neukonzeption eines Modells zur nutzenorientierten Analyse und Bewertung*. Universität Stuttgart.

Forrester, Jay Wright (1997). "Industrial dynamics". In: *Journal of the Operational Research Society* 48.10, S. 1037–1041.

Freedman, David und Persi Diaconis (1981). "On the histogram as a density estimator: L 2 theory". In: *Probability theory and related fields* 57.4, S. 453–476.

Freeman, R Edward (2010). *Strategic management: A stakeholder approach*. Cambridge University Press.

Fromm, Thomas, Markus Balser und Klaus Ott (2016). "Diesel-Autos: Der Abgasskandal - ein Debakel für die gesamte Autoindustrie". In: *sueddeutsche.de*. ISSN: 0174-4917. URL: http://www.sueddeutsche.de/wirtschaft/abgasaffaere-die-abgasaffaere-ein-debakel-fuer-die-gesamte-autoindustrie-1.2961703 (besucht am 25. 01. 2017).

Gabler, Siegfried und Matthias Ganninger (2010). "Gewichtung". In: *Handbuch der sozialwissenschaftlichen Datenanalyse*. Springer, S. 143–164.

Genrich, H.J. und K. Lautenbach (1981). "System modelling with high-level Petri nets". In: *Theoretical Computer Science* 13.1, S. 109 –135. ISSN: 0304-3975. DOI: http://dx.doi.org/10.1016/0304-3975(81)90113-4. URL: http://www.sciencedirect.com/science/article/pii/0304397581901134.

Georgii, Hans-Otto (2015). *Stochastik: Einführung in die Wahrscheinlichkeitstheorie und Statistik*. Walter de Gruyter GmbH & Co KG.

Gilbert, Mr Dennis M (1995). "An examination of federal and commercial access control policy needs". In: *National Computer Security Conference, 1993 (16th) Proceedings: Information Systems Security: User Choices*. DIANE Publishing, S. 107.

Gillmann, Michael, Jeanine Weissenfels, Gerhard Weikum und Achim Kraiss (2000). "Performance and availability assessment for the configuration of distributed workflow management systems". In: *Advances in Database Technology—EDBT 2000*. Springer, S. 183–201.

Gleißner, Werner (2011). *Grundlagen des Risikomanagements im Unternehmen: Controlling, Unternehmensstrategie und wertorientiertes Management*. Vahlen.

Grosskopf, Alexander, Gero Decker und Mathias Weske (2009). *The process: business process modeling using BPMN*. Meghan Kiffer Press.

Günther, Christian W. und Wil M.P. van der Aalst (2007). "Fuzzy Mining - Adaptive Process Simplification Based on Multi-perspective Metrics". In: *Business Process*

Management, 5th International Conference, BPM 2007, Brisbane, Australia, September 24-28, 2007, Proceedings. Hrsg. von Gustavo Alonso, Peter Dadam und Michael Rosemann. Bd. 4714. Lecture Notes in Computer Science. Springer, S. 328–343. ISBN: 978-3-540-75182-3.

Günther, Christian W. und Anne Rozinat (2012). "Disco: Discover Your Processes". In: *Proceedings of the Demonstration Track of the 10th International Conference on Business Process Management (BPM 2012), Tallinn, Estonia, September 4, 2012*. Hrsg. von Niels Lohmann und Simon Moser. Bd. 940. CEUR Workshop Proceedings. CEUR-WS.org, S. 40–44.

Günther, Christian W und HMW Verbeek (2014). *XES-standard definition*. BPMcenter.org.

Hammer, Michael und James Champy (2009). *Reengineering the Corporation: Manifesto for Business Revolution, A*. Zondervan.

Hendricks, Kevin B und Vinod R Singhal (2005). "An empirical analysis of the effect of supply chain disruptions on long-run stock price performance and equity risk of the firm". In: *Production and Operations management* 14.1, S. 35–52.

Henze, Norbert (2013). *Stochastik für Einsteiger. Eine Einführung in die faszinierende Welt des Zufalls*. Bd. 10. Springer Spektrum.

Herley, Cormac (2009). "So Long, and No Thanks for the Externalities: The Rational Rejection of Security Advice by Users". In: *Proceedings of the 2009 Workshop on New Security Paradigms Workshop*. NSPW '09. Oxford, United Kingdom: ACM, S. 133–144. ISBN: 978-1-60558-845-2. DOI: 10.1145/1719030.1719050. URL: http://doi.acm.org/10.1145/1719030.1719050.

Hirsch-Kreinsen, Hartmut (2014). *Wandel von Produktionsarbeit - Ïndustrie 4.0"*. TU Dortmund.

Holderer, Julius, Rafael Accorsi und Günter Müller (2015). "When Four-eyes Become Too Much: A Survey on the Interplay of Authorization Constraints and Workflow Resilience". In: *Proceedings of the 30th Annual ACM Symposium on Applied Computing*. SAC '15. Salamanca, Spain: ACM, S. 1245–1248. ISBN: 978-1-4503-3196-8. DOI: 10.1145/2695664.2699497. URL: http://doi.acm.org/10.1145/2695664.2699497.

Holderer, Julius, Josep Carmona Vargas und Günter Müller (2016). "Security-sensitive tackling of obstructed workflow executions". In: *Proceedings of the International Workshop on Algorithms & Theories for the Analysis of Event Data 2016: Torun, Poland, June 20-21, 2016*. CEUR-WS. org, S. 126–137.

Holterman, Reint Jan (2013). *5 Common Pitfalls in Process Optimization*. Bonitasoft.

Howard, Ronald A (2012). *Dynamic probabilistic systems: Markov models*. Bd. 1. Courier Corporation.

Hubertus, Andreae (2008). *Die 9 Erfolgsfaktoren von Unternehmen, die in Deutschland wettbewerbsfähig produzieren*. URL: http://www.elektronikpraxis.vogel.de/themen/elektronikmanagement/strategieunternehmensfuehrung/articles/122337/ (besucht am 31. 01. 2017).

Hungenberg, Harald und Torsten Wulf (2015). *Grundlagen der Unternehmensführung: Einführung für Bachelorstudierende*. Springer-Verlag.

Häggström, Olle (2001). "Finite Markov Chains and Algorithmic Applications". In: *London Mathematical Society Student Texts*. Cambridge University Press.

Jablonski, Stefan und Christoph Bussler (Sep. 1996). *Workflow Management: Modeling Concepts, Architecture and Implementation*. International Thomson Computer Press. ISBN: 1850322228.

Jensen, Kurt (1991). "Advances in Petri Nets 1990". In: Hrsg. von Grzegorz Rozenberg. Berlin, Heidelberg: Springer Berlin Heidelberg. Kap. Coloured petri nets: A high level language for system design and analysis, S. 342–416. ISBN: 978-3-540-46369-6. DOI: 10.1007/3-540-53863-1_31. URL: http://dx.doi.org/10.1007/3-540-53863-1_31.

Jensen, Kurt, Søren Christensen und Lars M Kristensen (2006). "CPN Tools State Space Manual". In: *Department of Computer Science, Univerisity of Aarhus*.

Johansson, Henry J (1993). *Business process reengineering: Breakpoint strategies for market dominance*. John Wiley & Sons.

Kalibatiene, Diana und Olegas Vasilecas (2016). "Dynamic Business Process Simulation - A Rule- and Context- Based Approach". In: *Information and Software Technologies: 22nd International Conference, ICIST 2016, Druskininkai, Lithuania, October 13-15, 2016, Proceedings*. Hrsg. von Giedre Dregvaite und Robertas Damasevicius. Cham: Springer International Publishing, S. 199–207. ISBN: 978-3-319-46254-7. DOI: 10.1007/978-3-319-46254-7_16. URL: http://dx.doi.org/10.1007/978-3-319-46254-7_16.

Kaplan, Robert S und Steven R Anderson (2004). "Time-driven Activity-based Costing". In: *Harvard business review* 82.11, S. 131.

Kaspersky Lab (2016). *Measuring Financial Impact of IT Security on Businesses, IT Security Risks Report 2016*.

Kita, Hajime, Kazuhisa Taniguchi und Yoshihiro Nakajima (2016). *Realistic Simulation of Financial Markets*.

Kluza, K., K. Jobczyk, P. Wi?niewski und A. Lig?za (2016). "Overview of time issues with temporal logics for Business Process Models". In: *2016 Federated Conference on Computer Science and Information Systems (FedCSIS)*, S. 1115–1123.

Koslowski, Thomas G. (2014). "Resilience Management Information Systems". Diss.

Koulopoulos, Thomas M (1995). *The workflow imperative: building real world business solutions*. John Wiley & Sons, Inc.

Krafcik, John F (1988). "Triumph of the lean production system". In: *MIT Sloan Management Review* 30.1, S. 41.

Krahl, D. (2001). "The Extend simulation environment". In: *Simulation Conference, 2001. Proceedings of the Winter*. Bd. 1, 217–225 vol.1. DOI: 10.1109/WSC.2001.977270.

Krämer, Stefanie (2007). *Total Cost of Ownership: Konzept, Anwendung und Bedeutung im Beschaffungsmanagement deutscher Industrieunternehmen*. VDM Publishing.

Krause, Hans-Ulrich und Dayanand Arora (2010). *Controlling-Kennzahlen-Key Performance Indicators: Zweisprachiges Handbuch Deutsch/Englisch-Bi-lingual Compendium German/English*. Walter de Gruyter.

Lee, Hau L, Venkata Padmanabhan und Seungjin Whang (1997). "The bullwhip effect in supply chains". In: *MIT Sloan Management Review* 38.3, S. 93.

Li, JianQiang, YuShun Fan und MengChu Zhou (2004). "Performance modeling and analysis of workflow". In: *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans* 34.2, S. 229–242. ISSN: 1083-4427. DOI: 10.1109/TSMCA.2003.819490.

Lima Bezerra, Fábio de, Jacques Wainer und Wil M.P. van der Aalst (2009). "Anomaly Detection Using Process Mining". In: *BMMDS/EMMSAD*. Hrsg. von Terry A. Halpin, John Krogstie, Selmin Nurcan, Erik Proper, Rainer Schmidt, Pnina Soffer und Roland Ukor. Bd. 29. Lecture Notes in Business Information Processing. Springer, S. 149–161. ISBN: 978-3-642-01861-9.

Marsan, M Ajmone, A Bobbio, G Conte und A Cumani (1984). "Performance analysis of degradable multiprocessor systems using generalized stochastic Petri nets". In: *IEEE Computer Society Newsletters* 6.SI-1, S. 47–54.

Mayer, Richard J., Ph. D, Christopher P. Menzel, Ph. D, Michael K. Painter, Paula S. Dewitte, Ph. D, Human Resources Directorate, Richard J. Mayer, Ph. D. Michael, K. Painter, Christopher P. Menzel, Ph. D. Benjamin Perakath und Ph. D (1992). *IICE IDEF3 process description capture method report (al/tr-1992-0057*. Techn. Ber. Air Force Systems Command, Wright-Patterson Air Force.

Mayr, Ernst W (1984). "An algorithm for the general Petri net reachability problem". In: *SIAM Journal on computing* 13.3, S. 441–460.

Mercuri, Rebecca T. (Juni 2003). "Analyzing Security Costs". In: *Communications of the ACM* 46.6, S. 15–18. ISSN: 0001-0782. DOI: 10.1145/777313.777327. URL: http://doi.acm.org/10.1145/777313.777327.

Merlin, Philip M und David J Farber (1976). "Recoverability of communication protocols–Implications of a theoretical study". In: *Communications, IEEE Transactions on* 24.9, S. 1036–1043.

Molloy, Michael Karl (1981). "On the integration of delay and throughput measures in distributed processing models". Diss.

— (1982). "Performance Analysis Using Stochastic Petri Nets". In: *IEEE Transactions on Computers* C-31.9, S. 913–917. ISSN: 0018-9340. DOI: 10.1109/TC.1982.1676110.

Morin, Marie-Laure und Christine Vicens (2001). "Redundancy, business flexibility and workers' security: Findings of a comparative European survey". In: *International Labour Review* 140.1, S. 45–67.

Muehlen, Michael zur und Jan Recker (2008). "How Much Language Is Enough? Theoretical and Practical Use of the Business Process Modeling Notation". In: *Advanced Information Systems Engineering: 20th International Conference, CAiSE 2008 Montpellier, France, June 16-20, 2008 Proceedings*. Hrsg. von Zohra Bellahsène und Michel Léonard. Berlin, Heidelberg: Springer Berlin Heidelberg, S. 465–479. ISBN: 978-3-540-69534-9. DOI: 10.1007/978-3-540-69534-9_35. URL: http://dx.doi.org/10.1007/978-3-540-69534-9_35.

Murata, Tadao (1989). "Petri nets: Properties, analysis and applications". In: *Proceedings of the IEEE* 77.4, S. 541–580.

Nüttgens, Markus und Frank J Rump (2002). "Syntax und Semantik Ereignisgesteuerter Prozessketten (EPK)." In: *Promise*. Bd. 2, S. 64–77.

Özyürek, Hamide und Yusuf Dinç (2014). "Time-Driven Activity Based Costing". In: *International Journal of Business and Management Studies* 6.1, S. 97–117.

Papagiannakopoulou, Eugenia I, Maria N Koukovini, Georgios V Lioudakis, Nikolaos L Dellas, Dimitra I Kaklamani und Iakovos S Venieris (2015). "Privacy-aware access control". In: *Encyclopedia of Information Science and Technology, Third Edition*. IGI Global, S. 4403–4411.

Park, Jaehong und Ravi Sandhu (2002). "Towards usage control models: beyond traditional access control". In: *Proceedings of the seventh ACM symposium on Access control models and technologies*. ACM, S. 57–64.

Parmenter, David (2015). *Key performance indicators: developing, implementing, and using winning KPIs*. John Wiley & Sons.

Perrow, Charles (1994). "The Limits of Safety: The Enhancement of a Theory of Accidents". In: *Journal of Contingencies and Crisis Management* 2.4, S. 212–220. ISSN: 1468-5973. DOI: 10.1111/j.1468-5973.1994.tb00046.x. URL: http://dx.doi.org/10.1111/j.1468-5973.1994.tb00046.x.

Peterson, James L. (Sep. 1977). "Petri Nets". In: *ACM Comput. Surv.* 9.3, S. 223–252. ISSN: 0360-0300. DOI: 10.1145/356698.356702. URL: http://doi.acm.org/10.1145/356698.356702.

Petri, Carl Adam (1962). "Kommunikation mit automaten". In:

Pfluger, Bettina (2005). "Unser Lager ist die Straße". In: *DER STANDARD*. URL: http://derstandard.at/2170971/Unser-Lager-ist-die-Strasse (besucht am 06. 10. 2016).

Pieth, Mark (2011). *Anti-Korruptions-Compliance: Praxisleitfaden für Unternehmen*. Dike.

PwC, PricewaterhouseCoopers International Limited (2016). *Global Economic Crime Survey 2016*.

Ramchandani, C. (1974). *Analysis of asynchronous concurrent systems by timed Petri nets*. Techn. Ber. Cambridge, MA, USA.

Rappaport, Alfred (1986). *Creating shareholder value: the new standard for business performance*.

Ratzer, Anne Vinter, Lisa Wells, Henry Michael Lassen, Mads Laursen, Jacob Frank Qvortrup, Martin Stig Stissing, Michael Westergaard, Søren Christensen und Kurt Jensen (2003). "CPN Tools for Editing, Simulating, and Analysing Coloured Petri Nets". In: *Applications and Theory of Petri Nets 2003: 24th International Conference, ICATPN 2003 Eindhoven, The Netherlands, June 23–27, 2003 Proceedings*. Hrsg. von Wil M.P. van der Aalst und Eike Best. Berlin, Heidelberg: Springer Berlin Heidelberg, S. 450–462. ISBN: 978-3-540-44919-5. DOI: 10.1007/3-540-44919-1_28. URL: http://dx.doi.org/10.1007/3-540-44919-1_28.

Reckenfelderbäumer, Martin (2013). *Entwicklungsstand und Perspektiven der Prozesskostenrechnung*. Springer-Verlag.

Reisig, Wolfgang (2012). *Petri nets: an introduction*. Bd. 4. Springer Science & Business Media.

Rezaie, K., B. Manouchehrabadi und S. N. Shirkouhi (2009). "Duration Estimation, a New Approach in Critical Chain Scheduling". In: *2009 Third Asia International Conference on Modelling Simulation*, S. 481–484. DOI: 10.1109/AMS.2009.67.

Russell, Nick, Wil M.P. van der Aalst, Arthur HM Ter Hofstede und David Edmond (2005). "Workflow resource patterns: Identification, representation and tool support". In: *Advanced Information Systems Engineering*. Springer, S. 216–232.

Sandhu, R. S. und P. Samarati (1994). "Access control: principle and practice". In: *IEEE Communications Magazine* 32.9, S. 40–48. ISSN: 0163-6804. DOI: 10.1109/35.312842.

Sandhu, Ravi S, Edward J Coyne, Hal L Feinstein und Charles E Youman (1996). "Role-based access control models". In: *Computer* 29.2, S. 38–47.

Schadschneider, Andreas, Wolfram Klingsch, Hubert Klüpfel, Tobias Kretz, Christian Rogsch und Armin Seyfried (2011). "Evacuation dynamics: Empirical results, modeling and applications". In: *Extreme Environmental Events*. Springer, S. 517–550.

Schneider, Fred B (2004). "Least privilege and more". In: *Computer Systems*. Springer, S. 253–258.

Schöneborn, Frank (2013). *Strategisches controlling mit system dynamics*. Springer-Verlag.

Schonenberg, Helen (2009). "Abstraction Techniques for Performance Analysis by Simulation". In:

Spath, Dieter, Oliver Ganschar, Stefan Gerlach, Moritz Hämmerle, Tobias Krause und Sebastian Schlund (2013). *Produktionsarbeit der Zukunft-Industrie 4.0*. Fraunhofer Verlag Stuttgart.

Sterman, John D John D (2000). *Business dynamics: systems thinking and modeling for a complex world*. HD30. 2 S7835 2000.

Stocker, Thomas und Frank Böhr (2013). "IF-Net: A Meta-Model for Security-Oriented Process Specification". In: *STM*. Hrsg. von Rafael Accorsi und Silvio Ranise. Bd. 8203. Lecture Notes in Computer Science. Springer, S. 191–206. ISBN: 978-3-642-41097-0.

Swegles, Scott (1997). "Business process modeling with SIMPROCESS". In: *Winter Simulation Conference*, S. 606–610.

Teixeira, Marcelo, Richardson Ribeiro, Cesar Oliveira und Ricardo Massa (2015). "A quality-driven approach for resources planning in Service-Oriented Architectures". In: *Expert Systems with Applications* 42.12, S. 5366 –5379. ISSN: 0957-4174. DOI: http://dx.doi.org/10.1016/j.eswa.2015.02.004. URL: http://www.sciencedirect.com/science/article/pii/S0957417415000974.

Tumay, Kerim (1995). "Business Process Simulation". In: *Proceedings of the 27th Conference on Winter Simulation*. WSC '95. Arlington, Virginia, USA: IEEE Computer Society, S. 55–60. ISBN: 0-7803-3018-8. DOI: 10.1145/224401.224421. URL: http://dx.doi.org/10.1145/224401.224421.

Van Dongen, Boudewijn F, AK Alves De Medeiros und Lijie Wen (2009). "Process mining: Overview and outlook of petri net discovery algorithms". In: *Transactions on Petri Nets and Other Models of Concurrency II*. Springer, S. 225–242.

Von Stackelberg, Silvia, Susanne Putze, Jutta Mülle und Klemens Böhm (2014). "Detecting data-flow errors in BPMN 2.0". In: *Open Journal of Information Systems (OJIS)* 1.2, S. 1–19.

Werner, Hartmut (2013). *Supply Chain Management: Grundlagen, Strategien, Instrumente und Controlling*. Springer-Verlag.

Weske, Mathias (2012). *Business Process Management - Concepts, Languages, Architectures, 2nd Edition*. Springer. ISBN: 978-3-642-28615-5.

White, Stephen A et al. (2004). "Business process modeling notation". In: *Specification, BPMI. org*.

Wildemann, Horst (2001). "Das Just-in-time-Konzept: Produktion und Zulieferung auf Abruf, 5". In: *Aufl., München*.

Wolter, Christian, Michael Menzel und Christoph Meinel (2008). "Modelling Security Goals in Business Processes." In: *Modellierung, LNI*. Bd. 127, S. 201–216.

Wooldridge, Michael (2009). *An introduction to multiagent systems*. John Wiley & Sons.

Zahoransky, Richard M., Thomas Koslowski und Rafael Accorsi (2014). "Toward Resilience Assessment in Business Process Architectures". In: *Computer Safety, Reliability, and Security: SAFECOMP 2014 Workshops: ASCoMS, DECSoS, DEVVARTS, ISSE, ReSA4CI, SASSUR. Florence, Italy, September 8-9, 2014. Proceedings*. Hrsg. von Andrea Bondavalli, Andrea Ceccarelli und Frank Ortmeier. Cham: Springer International Publishing, S. 360–370. ISBN: 978-3-319-10557-4. DOI: 10.1007/978-3-319-10557-4_39. URL: http://dx.doi.org/10.1007/978-3-319-10557-4_39.

Zahoransky, Richard M., Christian Brenig und Thomas Koslowski (Aug. 2015). "Towards a Process-Centered Resilience Framework". In: *2015 10th International Conference on Availability, Reliability and Security*, S. 266–273. DOI: 10.1109/ARES.2015.68.

Zahoransky, Richard M., Julius Holderer, Adrian Lange und Christian Brenig (2016). "Process Analysis as First Step Towards Automated Business Security". In: *European Conference on Information Systems* 24.

Zehbold, Cornelia (1996). "Entwicklungsschwerpunkte, Bedeutsame Ansätze und Anwendungsstand der Lebenszykluskostenrechnung". In: *Lebenszykluskostenrechnung*. Springer, S. 77–152.