ulm university universität
uulm

Fakultät für Ingenieurwissenschaften und Informatik
Institut für Datenbanken und Informationssysteme
in Kooperation mit der Daimler AG

Masterarbeit
im Studiengang Informatik

Durchgängige Modellierung von prozessorientierten Anwendungen mit BPMN 2.0

vorgelegt von

Jochen Schmitzl

Juni 2010

Erstgutachter: Prof. Dr. Manfred Reichert
Betreuer: Dr. Thomas Beuter, Daimler AG
Durchgeführt bei: Daimler AG

Kurzfassung

Um wettbewerbsfähig zu bleiben, müssen Unternehmen ihre Arbeitsabläufe ständig kontrollieren und verbessern. Dazu werden die Abläufe in Geschäftsprozessmodellen festgehalten. Diese werden anschließend analysiert und optimiert, mit der Zielsetzung Liege- und Bearbeitungszeiten zu verkürzen und die vorhandenen Ressourcen besser zu nutzen.

Eine weitere Verbesserung der Produktivität kann durch eine Unterstützung des Arbeitsablaufes durch IT-Systeme erreicht werden. Dazu wird das Prozessmodell weiter detailliert und formalisiert und in einem weiteren Schritt implementiert. Dabei werden zunehmend serviceorientierte Architekturen verwendet. Diese bilden Geschäftsprozesse mittels lose gekoppelter Einheiten (Services) ab. Ziel dieser Entkopplung ist es, die Flexibilität zu erhöhen. Teile des Geschäftsprozesses (Services) sollen in einem anderen Umfeld wiederverwendet werden können, um so Kosten zu sparen.

Die Modellierung des Geschäftsprozesses wird zu Beginn durch die Fachabteilung durchgeführt. Diese modelliert den Geschäftsprozess grob, ohne auf jedes Detail einzugehen. Dazu verwenden sie eine möglichst einfach verständliche Notation. Um den Geschäftsprozess durch IT-Systeme ausführen zu können, wird er anschließend von der IT-Abteilung detailliert und formalisiert. Dazu wird eine formale Notation benötigt. Die Modellierung sollte möglichst durchgängig gestaltet werden. Durchgängig heißt die Verwendung einer einheitlichen Notation, um eine Wiederverwendung der Modelle zu erreichen und die Kommunikation zwischen Fach- und IT-Abteilung zu verbessern. Die von der Fachabteilung erstellten Modelle können bei einer durchgängigen Modellierung direkt von der IT-Abteilung übernommen und formalisiert werden.

Bei der Modellierung eines Geschäftsprozesses müssen neben dem Ablauf der Arbeitsschritte noch weitere Informationen, wie etwa beteiligte Personen oder verwendete und generierte Daten, abgebildet werden.

In der vorliegenden Arbeit werden Ansätze zur durchgängigen Modellierung entwickelt und bewertet. Es werden Ansätze zur fachlichen Modellierung von Bearbeiterzuordnung, Daten, Ausnahmen und Flexibilitätsstellen im Prozessmodell entwickelt. Die Ansätze werden bewertet und jeweils ein Ansatz ausgewählt. Es wird gezeigt, wie die modellierten Informationen in einem weiteren Schritt detailliert und formalisiert werden können.

Die Ansätze werden jeweils auf ihre Realisierbarkeit mittels der *Business Process Model and Notation 2.0* (BPMN 2.0) geprüft. Die BPMN 2.0 ist eine neue Notation, die derzeit in einer Betaversion vorliegt. Sie wurde mit dem Ziel entwickelt, eine Notation zu schaffen, die so einfach ist, dass sie von der Fachabteilung verstanden wird. Gleichzeitig soll sie jedoch so formal sein, dass sie von der IT-Abteilung verwendet werden kann, um ein formales Modell des Geschäftsprozesses zu erstellen. Aufgrund dieser Eigenschaften eignet sie sich für eine durchgängige Modellierung.

In dieser Arbeit werden zudem vorhandene Werkzeuge evaluiert. Dabei wird untersucht, inwieweit die BPMN 2.0 bereits unterstützt wird und ob die vorgestellten Ansätze mit den Werkzeugen umgesetzt werden können.

Vorwort

Die vorliegende Arbeit entstand im Jahr 2010 als Masterarbeit bei der Daimler AG.

Mein Dank gilt Herrn Dr. Thomas Beuter, der die Anregung für diese Arbeit lieferte und sie seitens der Daimler AG betreute. Bei der Durchführung der Arbeit stand er mir mit Rat und Tat zur Seite und unterstützte mich von der Konzeption bis zur Korrektur der Arbeit.
Ein spezieller Dank gilt Herrn Dr. Thomas Bauer und Herrn Stephan Buchwald, die mir ebenfalls jederzeit hilfreich zur Seite standen.
Weiter möchte ich mich bei Herrn Prof. Dr. Manfred Reichert sowie Dr. Thomas Beuter für die Übernahme des Erst- und Zweitgutachtens bedanken.

Ferner gilt mein Dank meiner Familie und meinen Freunden, die mich in allen Phasen meines Studiums unterstützten.

Eigenständigkeitserklärung

Hiermit versichere ich, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die angegebenen Hilfsmittel verwendet habe. Sinngemäße Übernahmen aus anderen Werken sind als solche kenntlich gemacht und mit genauer Quellenangabe (auch aus elektronischen Medien) versehen.

Ulm, den 10. Juni 2010 Jochen Schmitzl

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

Ein Unternehmen hat das Ziel Produkte zu erstellen oder Dienstleistungen für seine Kunden zu erbringen. Um diese Leistungen zu generieren, muss eine Folge von Einzelaktivitäten schrittweise ausgeführt werden. Die Abfolge von Schritten zur Erreichung eines Zieles wird als Geschäftsprozess bezeichnet.

Um wettbewerbsfähig zu bleiben, müssen Unternehmen eine hohe Produktqualität bieten, flexibel auf Marktänderungen reagieren und gleichzeitig möglichst wirtschaftlich arbeiten. Um diese Ziele zu erreichen, ist es nötig, die Geschäftsprozesse zu erfassen, zu analysieren und zu optimieren. In einem weiteren Schritt können die Arbeitsabläufe mithilfe von IT-Systemen unterstützt oder automatisiert werden.

Das Identifizieren, Modellieren, Analysieren, Ausführen und Kontrollieren von Geschäftsprozessen wird als Geschäftsprozessmanagement (engl. Business Process Management, BPM) bezeichnet [Aal04]. Seit seinem Aufkommen in den 80er Jahren gewinnt es immer mehr an Bedeutung.

Um Kenntnisse über einen Geschäftsprozess zu erlangen, diesen zu analysieren und in einer späteren Phase in IT-Systeme umzusetzen, ist es nötig, den Geschäftsprozess darzustellen. Diese Tätigkeit wird als Geschäftsprozessmodellierung bezeichnet. Die Darstellung geschieht meist in einer grafischen Form. Die Modellierung der Geschäftsprozesse wird durch Mitarbeiter der Fachabteilungen vorgenommen, die die fachlichen Anforderungen definieren. Wünschen die Fachabteilungen eine IT gestützte Steuerung ihrer Geschäftsprozesse, so muss eine weitere Formalisierung der Modelle erfolgen. Diese wird in der Regel von den IT-Abteilungen in Abstimmung mit den Fachabteilungen übernommen [Rup07]. Dabei werden zunehmend serviceorientierte Architekturen (SOA) verwendet [Jos07, Erl05, BBP09]. Diese bilden Geschäftsprozesse mittels lose gekoppelten Einheiten (Services) ab. Ziel dieser Entkopplung ist es, die Flexibilität zu erhöhen. Änderungen in den fachlichen Anforderungen sollen schneller als bisher in den entsprechenden IT-Applikationen umgesetzt werden können.

Im Laufe der Zeit haben sich viele verschiedene Modellierungssprachen zur Geschäftsprozessmodellierung für unterschiedliche Zielgruppen entwickelt. Modellierungssprachen für Geschäftsprozesse mit der Zielgruppe Fachbereich legen ihren Schwerpunkt auf gute und einfache Verständlichkeit für die Fachbereiche. Darunter leidet jedoch die formale Ausdrucksmächtigkeit, die für eine Umsetzung des Geschäftsprozesses in IT-Systeme benötigt wird. Modellierungssprachen, die hierauf ihren Schwerpunkt legen, sind dagegen aufgrund der Formalismen für Fachbereiche nicht gut und einfach verständlich. Um die Geschäftsprozesse zu automatisieren, werden sie aber von den IT-Abteilungen benötigt.

Im Sinne einer guten Kooperation zwischen Fach- und IT-Abteilungen und einer Wiederverwendbarkeit der Modelle ist es sinnvoll eine gemeinsame Modellierungssprache zu verwenden. Die *Business Process Model and Notation* (BPMN) versucht den Anforderung, die an eine gemeinsam genutzte Modellierungssprache gestellt werden, gerecht zu werden.

Die von der BPMN definierten Symbole und Verwendungsregeln sind einfach gehalten. Die Anzahl der wichtigsten Symbole ist gering und ihre Verwendung intuitiv. Durch diese Einfachheit kann sie von Fachabteilungen verwendet werden, um die fachlichen Anforderungen zu modellieren. Gleichzeitig bietet sie aber auch eine große Menge spezieller Symbole. Zudem können Detailinformationen des Geschäftsprozesses, die zur Ausführung benötigt werden, über die grafische Darstellung hinaus in Attributen beschrieben werden. Dadurch erreicht die BPMN die nötige Formalität, um von der IT-Abteilung zur Modellierung einer ausführbaren Applikation benutzt zu werden. Damit können Fach- und IT-Abteilungen die gleiche Modellierungssprache verwenden. Somit bietet die BPMN eine durchgängige Modellierung von der Aufnahme der Anforderungen in der Fachabteilung, bis hin zur Umsetzung in IT-Systeme durch die IT-Abteilung.

In diesem Kapitel wird in Abschnitt 1.1 die Aufgabenstellung dieser Masterarbeit beschrieben. Ein Anwendungsszenario, an dem sich die Beispiele in dieser Arbeit orientieren, wird in Abschnitt 1.2 aufgezeigt. In Abschnitt 1.3 werden die Anforderungen an die Modellierung von Geschäftsprozessen dargelegt. Den Abschluss bildet eine Erläuterung des Aufbaus der vorliegenden Arbeit.

1.1 Aufgabenstellung

In dieser Arbeit wird untersucht, wo die Stärken und Schwächen einer durchgängigen Modellierung mit der BPMN 2.0, die derzeit im Betastatus vorliegt und voraussichtlich im 2. Quartal 2010 erscheinen wird, liegen. Als Durchgängigkeit wird dabei eine integrierende, auf Wiederverwendung basierende Modellierung von der fachlichen Abteilung bis zum ausführbaren Modell verstanden.

Dazu werden die Anforderungen, die an die Geschäftsprozessmodellierung gestellt werden, bestimmt. Für die identifizierten Anforderungen wird allgmein eine Vorgehensweise zur Modellierung beschrieben. Es wird dabei jeweils die Umsetzung sowohl aus fachlicher, als auch aus IT Sicht betrachtet. Ein besonderes Augenmerk wird dabei auf die möglichst frühzeitige Modellierung von Informationen gelegt. So sollen in späteren Phasen des Entwicklungsprozesses Abstimmungsrunden zwischen IT und Fachbereich möglichst vermieden werden. Dieser Aspekt wird

im Folgenden *Frontloading* [End09] genannt. Durch die frühzeitige Modellierung und der Verwendung der BPMN 2.0 als gemeinsame Modellierungssprache soll das Business-IT-Alignment [Che08] verbessert werden.

Die vorgestellten Konzepte werden auf ihre Umsetzbarkeit mittels der Betaversion der BPMN 2.0 untersucht. Wo diese nicht gegeben ist, werden Erweiterungsmöglichkeiten vorgeschlagen.

Zum Abschluss werden auf dem Markt befindliche Modellierungswerkzeuge evaluiert. Dabei wird die Unterstützung der neuen BPMN 2.0, die Umsetzbarkeit der vorgestellten Modellierungskonzepte sowie die Benutzerfreundlichkeit der Werkzeuge untersucht.

1.2 Anwendungsszenario

In diesem Abschnitt wird ein Anwendungsszenario beschrieben, aus welchem Anforderungen an die Geschäftsprozessmodellierung abgeleitet werden können. Die Visualisierung erfolgt in einer frei gewählten, neutralen Notation.

Das in Abbildung 1.1 dargestellte Anwendungsszenario beschreibt den Prozess einer Autovermietungsfirma. Initiiert wird dieser Prozess durch den Eingang der Fahrzeugreservierung eines Kunden. Der Vermietungsvorgang beginnt mit dem Verarbeiten der Reservierung [A]. Anschließend folgt die Nutzung [B] des reservierten Fahrzeuges und zum Schluss wird eine Nachbereitung [C] durchgeführt. Die einzelnen Prozessschritte werden im Folgenden detaillierter beschrieben.

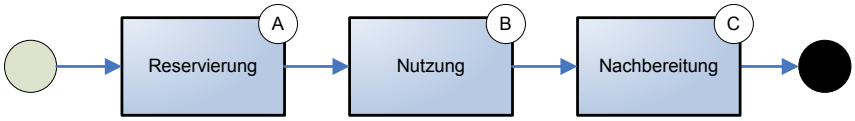


Abbildung 1.1: Übersicht des Prozesses einer Autovermietung

Abbildung 1.2 detailliert den Prozessschritt [A] "Reservierung" aus Abbildung 1.1. Die Reservierung eines Fahrzeuges beginnt damit, dass ein Sachbearbeiter aus der Abwicklungsabteilung die eingegangene Reservierung bearbeitet (siehe [A.1] Abb. 1.2). Dazu verwendet er eine Eingabemaske, welche im Grobdesign aus einer Altanwendung bereits vorliegt und mit dem Prozessschritt verknüpft werden soll. Nach der Bearbeitung liegt die Reservierung im Status "vorgeprüft" vor. Der Prozessschritt [A.2] "Kunde überprüfen" überprüft automatisch einen Kunden. Als Eingabe werden die Kundendaten benötigt, als Ausgabe liefert der Schritt eine Kundenbewertung, welche die für die nachfolgende Verzweigung benötigten Informationen liefert.

Die Verzweigung wählt einen von drei möglichen Pfaden. Schlägt die automatische Prüfung fehl, wird der Prozessschritt [A.1] "Reservierung bearbeiten" erneut ausgeführt. War die Prüfung erfolgreich kann die Kundenanfrage entweder abgelehnt (z.b. aufgrund fehlender Kreditwürdigkeit) und der Prozess beendet werden [A.3]. Andernfalls wird die Kundenanfrage angenommen und der Prozessschritt [A.4] 'Fahrzeug auswählen" kommt zur Ausführung. Dieser Schritt soll vom selben

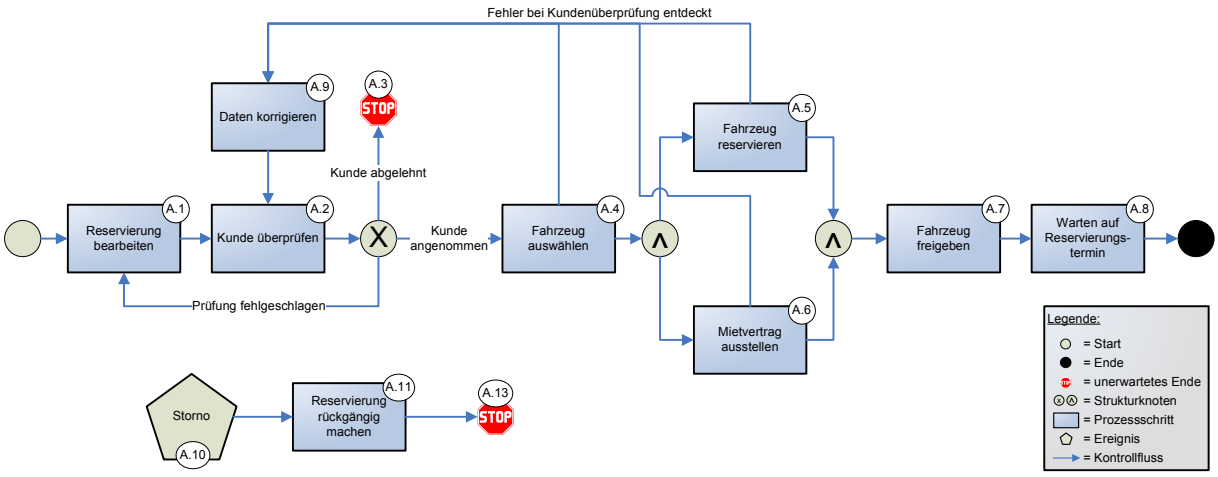


Abbildung 1.2: Detailprozess Reservierung

Sachbearbeiter ausgeführt, welcher bereits Prozessschritt A.1 ausgeführt hat. Als Eingabe benötigt der Prozessschritt die Reservierung im Status "vorgeprüft". Nachdem der Sachbearbeiter ein Fahrzeug ausgewählt hat, liegt die Reservierung im Status "bearbeitet" vor.

Die folgenden Schritte, A.5 "Fahrzeug reservieren" und A.6 "Mietvertrag ausstellen" werden parallel und vom selben Sachbearbeiter wie bereits Schritt A.1 ausgeführt. Beide Schritte benötigen als Eingabe die Reservierung im Status "bearbeitet" und die Kundendaten. Nach der durchgeführten Fahrzeugreservierung liegt die Reservierung im Status "durchgeführt" vor. Parallel dazu wird der Mietvertrag im Prozessschritt A.5 "Mietvertrag ausstellen" erstellt.

Während der Schritte A.4, A.5 und A.6 können durch den Sachbearbeiter Fehler bei der automatischen Kundenüberprüfung entdeckt werden. Ist dies der Fall, werden die aktuellen Prozessschritte beendet und die nach A.2 erfolgten Schritte rückgängig gemacht. Anschließend wird der Prozessschritt A.9 "Daten korrigieren" aufgerufen, welcher wiederum vom selben Sachbearbeiter wie die vorigen Schritte ausgeführt wird.

Wurde kein Fehler entdeckt, muss der Mietwagen noch freigegeben werden A.7. Dies erfolgt durch den Vorgesetzen des Sachbearbeiters, der den bisherigen Fall bearbeitet hat. Es folgt ein Prozessschritt, der lediglich auf den vereinbarten Reservierungstermin wartet A.10.

Während des gesamten Prozesses ist es jederzeit möglich, dass der Kunde die Reservierung durch eine Stornierung rückgängig macht. In diesem Fall empfängt der Prozess das Ereignis A.10 "Storno". Dies führt dazu, dass alle aktuell laufenden Prozessschritte abgebrochen werden. Es wird der Prozessschritt A.11 "Reservierung rückgängig machen" aufgerufen, welcher von einem Sachbearbeiter der Abwicklungsabteilung ausgeführt wird.

Ist der Prozessschritt A "Reservierung" abgeschlossen, wird der Prozessschritt B "Nutzung" ausgeführt. Dieser bearbeitet die Tätigkeiten von der Übergabe des Fahrzeuges an den Kunden bis zum Eingang der Zahlung. Abbildung 1.3 zeigt die Detaillierung des Prozessschrittes, die im Folgenden näher erläutert wird.

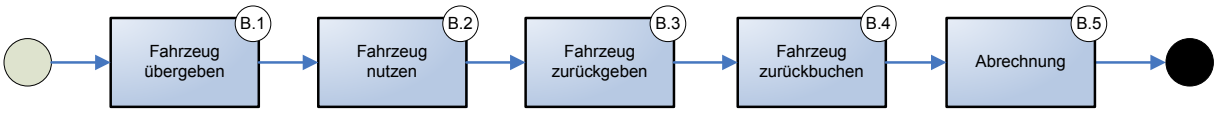


Abbildung 1.3: Detailprozess Nutzung

Zum vereinbarten Mietbeginn kommt der Fahrer zur Niederlassung, um das Auto abzuholen. Im Schritt B.1 "Fahrzeug übergeben" wird ihm das Fahrzeug durch einen Sachbearbeiter der entsprechenden Niederlassung übergeben. Der Fahrer nutzt das Fahrzeug über die vereinbarte Mietdauer B.2. Anschließend wird es von ihm zurückgegeben B.3. Dies geschieht indem der Fahrer das Auto in der vereinbarten Niederlassung abstellt und den Schlüssel in einen Briefkasten wirft. Durch einen Sachbearbeiter, der in der Reservierung als Rückgabestelle angegebenen Niederlassung wird das Fahrzeug im System als zurückgegeben verbucht B.4. Der anschließende Prozessschritt B.5 "Abrechnung" wird wie in Abb. 1.4 dargestellt weiter verfeinert.

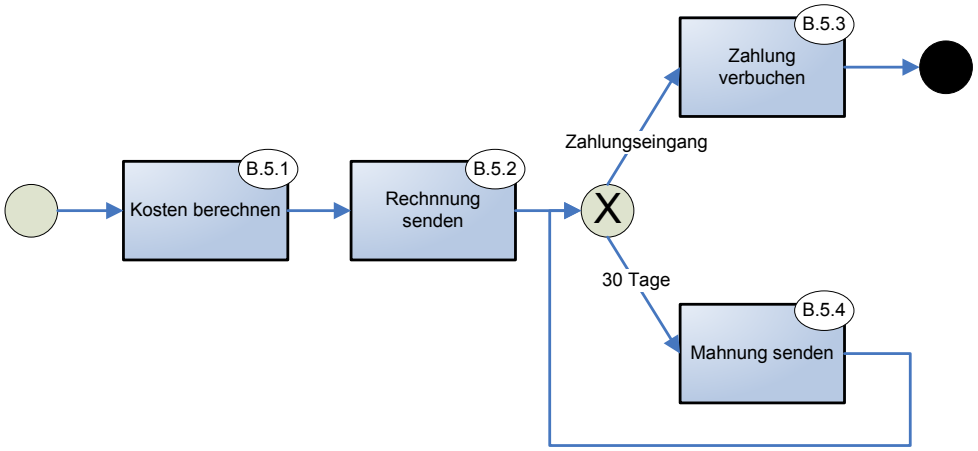


Abbildung 1.4: Sub-Prozess Abrechnung

Durch den Aufruf eines Services der Finanzabteilung werden im Prozessschritt B.5.1 "Kosten berechnen" die für die Miete anfallenden Kosten automatisch berechnet und eine Rechnung erstellt. Diese wird anschließend durch den Aufruf eines Services, den ebenfalls die Finanzabteilung anbietet, automatisch an den Kunden verschickt B.5.2. Der Kunde hat ab diesem Zeitpunkt 30 Tage Zeit die Forderung zu begleichen. Geht die Zahlung innerhalb dieser Frist nicht ein, wird der Prozessschritt B.5.4 "Mahnung versenden" aufgerufen, welcher einen Service des Post-Verwaltungs-Systems aufruft. Anschließend wird erneut auf den Zahlungseingang gewartet. Erfolgt der Zahlungseingang, wird er durch den Aufruf eines Services des Buchhaltungs-Systems B.5.3 verbucht. Die Abrechnung ist damit abgeschlossen.

Zur Optimierung der angebotenen Tarife und um das Kundenverhalten zu untersuchen findet eine "Nachbereitung" C statt. Diese ist in Abbildung 1.5 dargestellt. Im Prozessschritt C.1 "Auswahl treffen" wird automatisch per Zufallsprinzip ausgewählt, ob der Kunde überprüft werden soll oder nicht. Wird der Kunde nicht ausgewählt wird der Sub-Prozess beendet. Wird der Kunde ausgewählt, ermittelt Schritt C.2 alle Mietverträge des Kunden. Als Ergebnis liefert der

Prozessschritt eine Liste aller Mietvorgänge des Kunden. Die Anzahl der Mietvorgänge kann von Kunde zu Kunde variieren und kann zum Designzeitpunkt deshalb nicht angegeben werden.

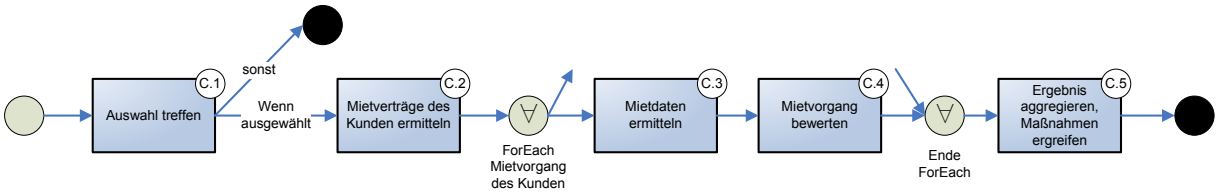


Abbildung 1.5: Detailprozess Nachbereitung

Die folgenden beiden Schritte C.3 und C.4 werden für jeden Mietvertrag des Kunden genau ein Mal bearbeitet. Der Kontrollfluss wird aufgespalten in genau so viele Teile, wie Mietverträge des Kunden vorliegen. Die Bearbeitung der einzelnen Verträge erfolgt parallel, es ist keine Synchronisation nötig. Erst wenn die Bewertung aller Verträge abgeschlossen ist, kann der nächste Prozessschritt gestartet werden [AHKB03].

Der Prozessschritt C.3 "Mietdaten ermitteln" ist ein automatischer Schritt, der durch einen Serviceaufruf realisiert wird. Der gerufene Service wird von der Kundenverwaltung angeboten. C.4 "Mietvorgang bewerten" wird manuell bearbeitet. Dies geschieht durch einen Sachbearbeiter der Niederlassung, in der das Auto zurückgegeben wurde. Um eine zeitnahe Bearbeitung zu gewährleisten, muss innerhalb von fünf Tagen ein Mitarbeiter diesen Prozessschritt zur Bearbeitung reserviert haben. Ist dies nicht der Fall wird die Bewertung automatisch dem Niederlassungsleiter dieser Niederlassung zugeordnet. Nach der Reservierung hat der Mitarbeiter drei Tage Zeit zur Bearbeitung. Ist der Schritt dann noch nicht abgeschlossen wird eine E-Mail an den Niederlassungsleiter geschickt. Im abschließenden Schritt C.5 werden die erhobenen Daten vom Kundenmanager aggregiert. Je nach Ergebnis der Auswertung werden eventuell Maßnahmen zur Verbesserung festgelegt.

Nachdem der Prozess von der Fachabteilung spezifiziert wurde, wird er der IT-Abteilung zur weiteren Bearbeitung übergeben. Um den Prozess in einem IT-System umzusetzen, genügen die Angaben der Fachabteilung noch nicht. Es sind einige Veränderungen und Verfeinerungen von der IT-Abteilung notwendig. Diese werden in Abstimmung mit der Fachabteilung vorgenommen.

Da die aufeinander folgenden Prozessschritte A.4, A.5, und A.6 immer von der gleichen Person bearbeitet werden, sollen sie zu einem einzigen Prozessschritt zusammengefasst werden und mit einer einzigen Eingabemaske realisiert werden. Das Design dieser Maske soll mit dem neuen Prozessschritt verknüpft werden.

Das Warten auf den Reservierungstermin soll mit dem Ziel realisiert werden, dass der zuständige Sachbearbeiter erst am Tage des Termins einen Eintrag in seiner Workliste zu sehen bekommt. Eine Worklist ist eine Art ToDo-Liste, die einem Mitarbeiter anzeigt, welche Aufgaben er zu erledigen hat [BP05].

Die Prozessschritte B.2 "Fahrzeug nutzen" und B.3 "Fahrzeug zurückgeben" können nicht durch die IT unterstützt werden. Prozessschritt B.4 "Fahrzeug zurückbuchen" wird deshalb durch ein Event "Fahrzeug zurückgegeben" gestartet. Die Korrelation zu einer Prozessinstanz erfolgt über das Kennzeichen des zurückgegebenen Fahrzeuges.

Nach erfolgreicher Bearbeitung des Schrittes wird der Sub-Prozess B.5 "Abrechnung" aufgerufen. Dieser Aufruf läuft vollständig automatisch, das heißt ohne Benutzerinteraktion, ab. Aufgrund der möglichen langen Wartezeit auf den Zahlungseingang soll der Sub-Prozess unterbrechbar (transaktional) sein. Bevor dieser Prozessschritt gestartet werden kann, müssen alle Daten vollständig vorliegen (Mietvertrag, Kundendaten insb. die Kundenadresse).

Alle Prozessschritte des Sub-Prozesses B.5 sind automatisiert und werden als Service-Aufruf realisiert. Der jeweilige Service und der passende Aufruf müssen durch die IT genau spezifiziert werden, damit der Service-Aufruf automatisch durchgeführt werden kann. Um die Daten in die von den jeweiligen Services benötigten Formate zu überführen, ist eine Datentransformation (Mapping) erforderlich. Unter anderem müssen die Felder Nachname und Vorname in ein gemeinsames Feld Name überführt werden, das Feld Ort in Wohnort umbenannt und das Feld Bankverbindung in Kontonummer, BLZ und Kontoinhaber aufgespalten werden. Um diese Transformationen auszuführen ist es notwendig, einen zusätzlichen, automatischen Prozessschritt einzufügen.

Um eine spätere Ausführung des Prozesses zu ermöglichen, ist es notwendig alle relevanten Informationen möglichst formal zu beschreiben. Dazu gehört zum Beispiel die Wartezeit auf eine eingehende Zahlung, deren Ende per Event festgestellt werden kann. Gleiches gilt für die Überschreitung der 30-Tage-Frist, nach der eine Mahnung versendet wird.

Bei der Nachbereitung soll die Auswahl, ob der Kunde überprüft wird oder nicht, nicht rein zufällig erfolgen sondern durch eine Business-Rule realisiert werden. Es werden lediglich Kunden überprüft, die mindestens fünf Mietvorgänge vorweisen und eine durchschnittliche Mietdauer von zwei Tagen überschreiten. Zusätzlich soll die Regel eine Zufallskomponente enthalten. Auch wenn der Kunde die genannten Bedingungen erfüllt, wird er nur mit einer 20%-igen Wahrscheinlichkeit ausgewählt. Das anschließende foreach-Konstrukt soll grafisch dargestellt werden können. Es soll erkennbar sein, wie sich die Anzahl der parallelen Zweige zusammensetzt und welche Daten in jeweils einen Pfad einfließen.

1.3 Anforderungen

Dieser Abschnitt beschreibt Anforderungen an die Prozessmodellierung. Die Anforderungen ergeben sich aus dem in Abschnitt 1.2 vorgestellten Anwendungsszenario. Die Anforderungen werden beschrieben und deren Relevanz durch Beispiele aus dem Anwendungsszenario deutlich gemacht. Anschließend werden die Anforderungen verschiedenen Kategorien zugeordnet.

Anforderung 1: Ereignisse
Die Ausführung eines Prozesses wird oft durch Ereignisse beeinflusst. Neben intern getriggerten Ereignissen, wie zum Beispiel zeitlichen Ereignissen, können Ereignisse auch von außen getriggert werden. Ein Beispiel dafür ist der Eingang einer Stornierung A.10. Dabei ist es wichtig externe Ereignisse der richtigen Prozessinstanz zuzuordnen.

Anforderung 2: Vorbedingungen
Um einen Prozessschritt starten zu können müssen häufig Vorbedingungen erfüllt sein. Insbesondere das Vorhandensein der benötigten Eingabedaten ist erforderlich.

Anforderung 3: Zeitanforderung
Zeit ist ein wichtiger Aspekt bei der Ausführung von Geschäftsprozessen. Häufig müssen Deadlines modelliert werden oder die verstrichene Zeit zwischen zwei Prozessschritten gemessen werden. Im Anwendungsszenario wird zum Beispiel dem Kunden eine Mahnung geschickt, nachdem er 30 Tage lang nicht auf die Rechnung reagiert.

Anforderung 4: komplexer Kontrollfluss
Der Kontrollfluss soll sich nicht auf einfache, sequentielle Abläufe beschränken. Auch kompliziertere Konstrukte, wie bedingte Verzweigungen, Parallelisierung, Rücksprünge etc. sollen modelliert werden können.

Anforderung 5: komplexe Bedingungen
Neben dem üblichen Kontrollfluss soll es möglich sein, komplexe Bedingungen zu modellieren. Zum Beispiel sollen zeitliche Bedingungen (nach 30 Tagen) oder Kontrollflüsse, die durch im Prozesskontext vorhandenen Daten bedingt sind, modelliert werden können.

Anforderung 6: foreach-Konstrukt
Um eine unbekannte Menge an Daten zu verarbeiten muss die Möglichkeit vorhanden sein einen Prozessschritt beziehungsweise mehrere Prozessschritte beliebig oft ausführen zu können.

Anforderung 7: Bearbeiterzuordnung
Prozessschritte können automatisch oder durch das Eingreifen von menschlichen Bearbeitern ausgeführt werden. Wird ein Bearbeiter für die Ausführung eines Schrittes benötigt, muss spezifiziert werden, welche Eigenschaften dieser Bearbeiter erfüllen muss. Nur Mitarbeiter, die diese Eigenschaften erfüllen, werden dem Prozessschritt zugeordnet und können diesen ausführen. Der Prozessschritt A.1 "Reservierung bearbeiten" muss zum Beispiel durch einen Sachbearbeiter aus der Abwicklungsabteilung bearbeitet werden. Komplexere Zuordnungen, die über die reine Angabe einer Organisationseinheit hinausgehen, werden ebenfalls benötigt. Der Prozessschritt A.4 "Fahrzeug auswählen" soll zum Beispiel vom gleichen Bearbeiter bearbeitet werden wie der Prozessschritt A.1 "Reservierung bearbeiten".

Anforderung 8: weitere Beteiligte
Neben der Person, die einen Prozessschritt ausführt, können auch noch weitere Beteiligte in die Bearbeitung involviert sein. Im Prozessschritt B.1 "Fahrzeug übergeben" ist zum Beispiel neben dem Mitarbeiter zusätzlich auch der Kunde involviert.

Anforderung 9: Datenobjekte
Beim Ablauf eine Geschäftsprozesses werden von den einzelnen Prozessschritten Daten verwendet und generiert. Im Anwendungsszenario ist dies zum Beispiel die Rechnung. Diese müssen in den Fachabteilungen identifiziert und modelliert werden.

Anforderung 10: Eingabe- und Ausgabedaten
Für jeden Prozessschritt muss festgelegt werden können, welche Daten als Eingabe benötigt werden und welche Daten ausgegeben werden. Der Prozessschritt B.5.2 "Rechnung senden" benötigt zum Beispiel zum Berechnen der Kosten des Vertrages die Kundendaten und den Mietvertrag als Eingabe und generiert daraus eine Rechnung.

Anforderung 11: komplexe Datenobjekte
Neben einfachen Daten sollen auch komplexe Datentypen wie etwa Listen oder Arrays definiert

werden können. Im Anwendungsszenario wird bei der Nachbereitung C zum Beispiel eine Liste aller Mietverträge eines Kunden benötigt.

Anforderung 12: fachliche Ausnahmen
In Geschäftsprozessen treten immer wieder Ausnahmen auf, die vom Fachbereich erwartet werden, sogenannte fachliche Ausnahmen. Zum Beispiel könnte der Kunde die Reservierung stornieren A.10. Um auf diese Ausnahmen reagieren zu können, müssen diese explizit ausmodelliert werden.

Anforderung 13: technische Ausnahmen
Bei der Ausführung eines Geschäftsprozesses können technische Ausnahmen auftreten. Dies sind Ausnahmen, die nicht erwartet wurden und von der Technik verursacht werden. Sie werden meist erst von der IT-Abteilung erkannt und modelliert. Schlägt zum Beispiel der Aufruf des Web-Services in Prozessschritt B.5.1 fehl, ist dies eine technische Ausnahme. Potentielle technische Ausnahmen müssen abgefangen und adäquat behandelt werden können.

Anforderung 14: Transaktionen
Mehrere Prozessschritte sollen zu einer Transaktion zusammengefasst werden können. Eine Transaktion ist eine Folge von Prozessschritten, die entweder alle oder gar nicht ausgeführt werden [Pap03]. Tritt ein Fehler auf, nachdem bereits ein Teil der Prozessschritte in der Transaktion ausgeführt wurden, müssen diese Schritte zurückgesetzt werden. Wird zum Beispiel die Reservierung storniert A.10 muss ein schon reserviertes Fahrzeug wieder freigegeben werden.

Anforderung 15: Maskendesign
Eine Implementierung von Benutzerschnittstellen sollte sich möglichst stark an den Anforderungen der Fachbereiche orientieren. Nur wenn dies der Fall ist, wird die Schnittstelle von den Benutzern akzeptiert. Um dies zu gewährleisten, ist eine möglichst frühe, mit den Fachbereichen abgestimmte Modellierung der Benutzerschnittstellen notwendig. Im Prozessschritt A.1 "Reservierung bearbeiten" sollte es zum Beispiel möglich sein, einen Screenshot der Oberfläche der Altanwendung zu verknüpfen.

Anforderung 16: Geschäftsregeln
Um die Geschäftslogik innerhalb eines Geschäftsprozesses abbilden zu können ist es nötig Regeln zu beschreiben. Diese Geschäftsregeln können direkt im Prozessmodell beschrieben werden oder durch die Verknüpfung mit einer Business-Rule-Engines realisiert werden. Im Anwendungsszenario ist zum Beispiel denkbar, dass unter gewissen Umständen eine Schufaauskunft eingeholt wird, um die Kreditwürdigkeit eines Kunden zu prüfen. Die Regeln, wann eine Auskunft eingeholt wird und wann nicht, können beliebig komplex sein.

Anforderung 17: Frontloading
Um möglichst kostengünstig und schnell ein ausführbares Modell des Geschäftsprozesses entwickeln zu können, ist es essentiell, möglichst viele ausführungsrelevante Informationen in einer frühen Phase in das Geschäftsprozessmodell aufzunehmen. Dies gilt für Informationen wie zum Beispiel, dass der Prozessschritt A.2 "Kunde überprüfen" ein automatischer Prozessschritt ist. Dabei gilt generell der Grundsatz, je mehr Informationen früh modelliert werden, desto besser. Dadurch wird die in späteren Phasen nötige Abstimmung zwischen IT und Fachabteilung minimiert, da das gesamte Wissen des Fachmodellierers bereits in dieser frühen Phase in das Modell integriert wurde. Trotz des Anspruchs viele Informationen möglichst früh zu modellieren, sollte der Modellierer aus dem Fachbereich nicht überfordert werden.

Anforderung 18: Flexibilität
Ein Geschäftsprozess kann sich im Laufe der Zeit ändern, er ist flexibel. Die Fachabteilung kann durch Erfahrungen aus der Vergangenheit sagen, welche Stellen davon betroffen sind. Solche Stellen, an denen es wahrscheinlich ist, dass eine Änderung zu einem späteren Zeitpunkt auftreten wird, sollten markiert werden. Die Realisierung solcher Flexibilitätsstellen sollte so erfolgen, dass spätere Änderungen im Geschäftsprozess leicht umgesetzt werden können. Wird zum Beispiel zum Prüfen der Liquidität eines Kunden ein Web-Service der Schufa aufgerufen, dessen Aufruf sich häufig ändert, kann dies im Modell vermerkt werden.

Anforderung 19: Formalität
Um einen Geschäftsprozess ausführbar zu machen, müssen alle für die Ausführung relevanten Sachverhalte formal beschrieben werden. Ein Freitext "geht nach 30 Tagen keine Zahlung ein, diesen Weg wählen" kann durch eine Process-Engine nicht verstanden werden. Deshalb muss die Angabe formalisiert werden und in ein standardisiertes Format gebracht werden, das von einer Process-Engine verstanden wird.

Zusammenfassung

Die identifizierten Anforderungen können, wie in Abbildung 1.6 gezeigt, verschiedenen Kategorien zugeordnet werden. Ein Teil der Anforderungen betrifft den Kontrollfluss des Geschäftsprozesses direkt. Weitere Anforderungen werden an die Modellierung von bei der Prozessausführung beteiligten Daten und Organisationseinheiten gestellt. Zudem müssen Ausnahmen im Ablauf des Geschäftsprozesses modelliert werden können. Darüber hinaus existieren weitere Anforderungen, die nicht mehr zusammengefasst werden können. Die Anforderungen A17 Frontloading, A18 Flexibilität und A19 Formalität betreffen jeweils alle Aspekte der Modellierung und sind deshalb vertikal eingezeichnet.

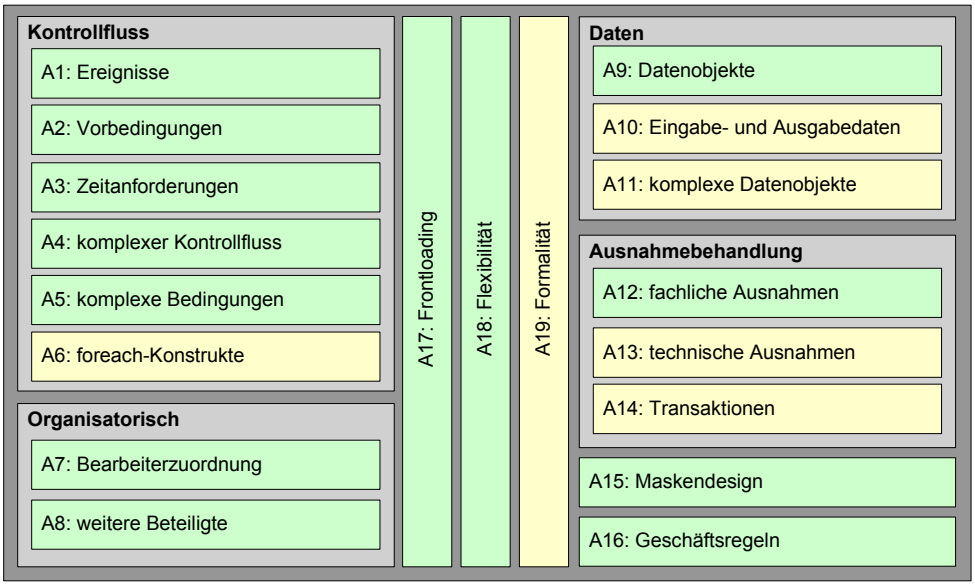


Abbildung 1.6: Übersicht der Anforderungen bei der Prozessmodellierung

Die farbliche Unterscheidung zeigt den Zeitpunkt der Modellierung der jeweiligen Anforderungen. Die grün hinterlegten Anforderungen müssen schon früh in der Modellierung beachtet werden, sind also bereits für die Fachabteilungen relevant. Die gelb hinterlegten Anforderungen werden erst durch die IT-Abteilungen bearbeitet und sind folglich erst in späteren Phasen der Modellierung relevant.

Die aufgezeigten Anforderungen an die Prozessmodellierung müssen von den Modellierern umgesetzt werden. Dies kann von ihnen nur dann getan werden, wenn eine Modellierungssprache existiert, die die zur Modellierung der Anforderungen nötigen Konstrukte bereitstellt. Zudem muss eine Vorgehensweise zur Modellierung der Informationen beschrieben werden. Diese müssen zudem durch die vorhandenen Werkzeuge unterstützt werden.

1.4 Aufbau der Arbeit

In Kapitel 2 werden zunächst die Grundlagen vorgestellt. Es werden sowohl Vorgehensweisen bei der Prozessmodellierung, als auch Modellierungssprachen für die Geschäftsprozessmodellierung eingeführt. Kapitel 3 bietet eine detailliertere Einführung in die neue Modellierungssprache BPMN 2.0, die im Rahmen dieser Arbeit untersucht wird.

In Kapitel 4 und 5 werden Ansätze vorgestellt, um die Datenmodellierung, die Bearbeiterzuordnung, die Ausnahmebehandlung und die Markierung von Flexibilitätsstellen vornehmen zu können. In Kapitel 4 wird dabei auf die fachliche Modellierung eingegangen, Kapitel 5 beschäftigt sich mit der darauf folgenden Formalisierung der Modelle durch die IT-Abteilung. Die Ansätze werden jeweils auf ihre Realisierbarkeit mittels der BPMN 2.0 überprüft.

In Kapitel 6 werden kurz die Schritte skizziert, die nötig sind, um das Modell in eine Ausführungsumgebung zu überführen.

In Kapitel 7 werden auf dem Markt befindliche Werkzeuge untersucht, inwieweit sich mit ihnen die vorgestellten Ansätze umsetzen lassen.

Mit einer Diskussion (Kapitel 8) und einer abschließenden Zusammenfassung (Kapitel 9) endet diese Masterarbeit.

2 Grundlagen

In diesem Kapitel werden einige für die Arbeit relevante Aspekte eingeführt. In Abschnitt 2.1 werden allgemeine Vorgehensweisen bei der Prozessmodellierung und die Prozessmodellierung, wie sie bei der Daimler AG Anwendung findet, vorgestellt. In Abschnitt 2.2 werden verschiedene Modellierungssprachen für Geschäftsprozesse aufgezeigt.

2.1 Vorgehensweise bei der Prozessmodellierung

In diesem Abschnitt werden Vorgehensweisen für die Prozessmodellierung vorgestellt. In Abschnitt 2.1.1 wird als bekannteste Vorgehensweise, die Model Driven Architecture beschrieben. Der folgende Abschnitt 2.1.2 behandelt weitere Vorgehensweisen. Am Ende wird in Abschnitt 2.1.3 die Vorgehensweise bei der Prozessmodellierung in der Daimler AG beschrieben.

2.1.1 Model Driven Architecture

Die Model Driven Architecture (MDA) ist ein modellgetriebener Softwareentwicklungsansatz. Er wurde von der Object Management Group (OMG) standardisiert [OMG03] und basiert auf der ebenfalls von der OMG standardisierten UML [OMG09c]. Das Grundprinzip ist die Trennung von Funktionalität und Technik [Bro04]. Der Entwicklungsprozess wird in die Geschäfts- und die IT-Sicht unterteilt. Ziel ist es, plattformspezifische Modelle möglichst automatisch aus plattformunabhängigen Modellen abzuleiten. Dadurch soll der Aufwand für die Softwareentwicklung verringert und insbesondere die Anpassung an neue Technologien erleichtert werden.

Der Entwicklungsprozess der MDA ist in vier Ebenen unterteilt (siehe Abb. 2.1):

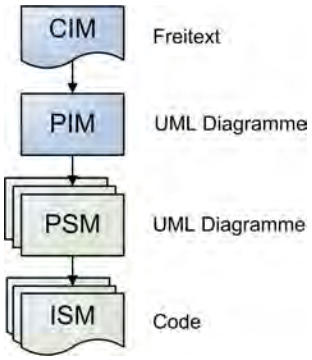


Abbildung 2.1: Ebenen der Model Driven Architecture

- **Computation Independent Model (CIM)**: Plattformunabhängige umgangssprachliche Beschreibung der fachlichen Anforderungen des Geschäftsprozesses
- **Platform Independent Model (PIM)**: Plattformunabhängiges Modell des Geschäftsprozesses
- **Platform Specific Model (PSM)**: Plattformabhängiges Modell des Geschäftsprozesses, das die Architektur beschreibt
- **Implementation Specific Model (ISM)**: IT-technische Realisierung des zuvor beschriebenen Geschäftsprozesses in Code

Der erste Schritt ist die technikunabhängige Modellierung der fachlichen Anforderungen in einer umgangssprachlichen Beschreibung. Nachdem alle Anforderungen im CIM aufgenommen wurden, wird daraus das PIM entwickelt. Die Transformation muss aufgrund der informellen Modellierung des CIMs manuell erfolgen. Das PIM wird in UML modelliert und bildet die Grundlage der weiteren Entwicklung. Dabei wird die Geschäftslogik ohne die Berücksichtigung einer späteren Implementierung definiert. Das PIM enthält nicht nur statische Informationen wie zum Beispiel die Darstellung von Use Case Diagrammen, sondern auch dynamische Aspekte zum Beispiel durch Aktivitätsdiagramme.

Wichtig bei der Beschreibung des PIM ist, dass dieses möglichst vollständig formal ist. Dadurch kann aus dem PIM durch eine automatische Transformation das PSM erstellt werden. Das so generierte PSM kann nun um plattformspezifische Informationen erweitert werden und so die auf die jeweilige Ziel-Umgebung angepasst werden. Als Ziel-Umgebung ist zum Beispiel .NET oder CORBA denkbar. Das PSM wird ebenfalls durch UML Diagramme beschrieben.

Aus dem erstellten PSM wird in einem weiteren Transformationsschritt das ISM generiert. Die Transformation in den Code der entsprechenden Ziel-Umgebung läuft automatisch ab. Der erstellte Code kann von einem Implementierer anschließend verändert werden.

Bei einer späteren Änderung des Ablaufs sollte diese Änderung möglichst nicht nur im Code durchgeführt werden, sondern in allen Ebenen Beachtung finden. Eine Änderung im PIM sollte dabei möglichst automatisch eine Änderung im PSM und im ISM nach sich ziehen, sodass alle Ebenen jeweils konsistent sind.

Durch die Aufteilung in die vier Modellebenen kann eine Trennung der Verantwortlichkeiten für die einzelnen Ebenen vorgenommen werden. Während das CIM und das PIM von den Fachabteilungen modelliert wird, ist für die Modellierung des PSM und des ISM die IT-Abteilung verantwortlich.

2.1.2 Weitere Vorgehensweisen

Neben der MDA gibt es noch weitere Vorgehensweisen zur Prozessmodellierung. Allen ist gemeinsam, dass eine Trennung von Geschäfts- und IT-Sicht vorgenommen wird. Aus der Vielzahl von Ansätzen werden im folgenden drei bekannte herausgegriffen und kurz erläutert. Alle sind von großen Herstellern spezifiziert und werden von deren jeweiligen Software unterstützt.

Service-Oriented Modeling and Architecture (SOMA) ist eine Entwicklungsmethode von IBM [AGA+08, Ars04]. Bei der SOMA-Methode existiert eine Trennung von Geschäfts- und IT-Sicht. Innerhalb dieser beiden Sichten werden einzelne Phasen durchlaufen, in denen genau spezifiziert ist, was zu tun ist. Die Methode wird durch das von IBM bereitgestellte Werkzeug Rational Software Architect unterstützt [IBM09].

M3 ist eine Entwicklungsmethode, die von MID definiert wurde. Sie baut auf der MDA auf [DPRW08]. M3 basiert ebenfalls auf der UML, die von MID durch UML Stereotypen spezifiziert wurde. Unterstützt wird die Methode durch den von MID angebotenen Innovator [MID10b].

Auch von IDS Scheer wird eine Enwicklungsmethodik angeboten, die ARIS Value Engineering (AVE) [IDS05]. Diese definiert einen Prozesslebenszyklus, der sich von der Sammlung der Informationen eines Geschäftsprozesses, über die Modellierung und die Implementierung bis hin zur Prozessüberwachung erstreckt. Die Schritte sollen nicht einmalig durchlaufen werden, sondern einer ständigen Verbesserung unterliegen und bei eingebrachten Änderungen von vorne durchlaufen werden. AVE wird durch die ARIS Design Platform unterstützt, welche zudem eine enge Kopplung mit SAP bietet [Sei06].

2.1.3 Prozessmodellierung bei der Daimler AG

Auch die Daimler AG arbeitet mit einer Entwicklungsmethodik, die auf der MDA beruht [BBR10b]. Abbildung 2.2 zeigt die Ebenen, die bei der Prozessmodellierung durchlaufen werden müssen. Die Geschäftsprozesse werden von den Fachabteilungen erfasst, die diese aus Anwendersicht betrachten. Sie erstellen ein sogenanntes fachliches Modell. Für eine spätere Software-Realisierung eines Geschäftsprozesses wird dieser durch die IT-Abteilung genauer spezifiziert, formalisiert und in dieser Form mit den Fachabteilungen abgestimmt. Die Ergebnisse werden im Systemmodell festgehalten. In einem weiteren Schritt wird das Systemmodell in ein ausführbares Modell überführt.

Die Modellierung des fachlichen Modells in den Fachabteilungen geschieht durch Endanwender beziehungsweise den Prozessverantwortlichen. Diese Personen haben in der Regel keinen IT-Hintergrund, weshalb die Dokumentation nicht sehr formal ist. Wert wird insbesondere auf gute Verständlichkeit gelegt, weshalb einfache grafische Notationen (z.B. EPK) und textuelle

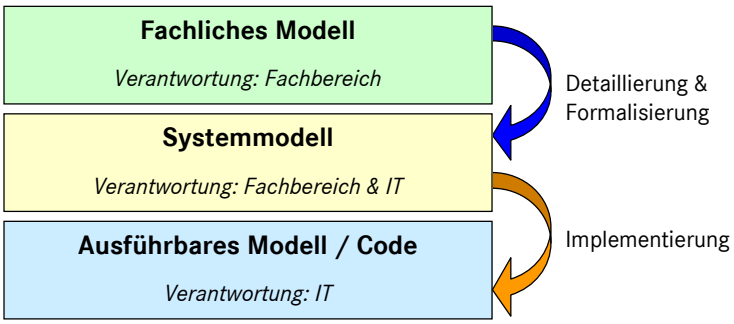


Abbildung 2.2: Ebenenmodell der Modellierungsmethodik bei der Daimler AG

Beschreibungen verwendet werden. Trotz der fehlenden Formalisierung werden in dieser frühen Phase schon alle relevanten Aspekte modelliert. Dazu zählt neben dem eigentlichen Ablauf des Geschäftsprozesses zum Beispiel die beteiligten Organisationseinheiten und die verwendeten Daten. Das Ziel ist ein möglichst vollständiges Prozessmodell zu bekommen, das trotzdem so einfach ist, dass es von der Fachabteilung verstanden wird. Der modellierte Prozess wird möglichst durch Analyse und Simulation auf Korrektheit und potentielle Verbesserungsmöglichkeiten untersucht.

Das im Fachbereich entstandene fachliche Modell wird als Grundlage für die weitere Entwicklung genommen. Es wird in das Systemmodell überführt und anschließend schrittweise weiter detailliert sowie formalisiert. Das bedeutet, dass Stellen, an denen das fachliche Modell nur textuelle Beschreibungen hat und noch offene Aspekte sind, beseitigt werden müssen. Jede durchgeführte Änderung sollte mit dem Fachbereich abgestimmt werden. Die Verantwortung für das Systemmodell liegt bei der IT-Abteilung, die jedoch ständig Rücksprache mit der Fachabteilung halten sollte. Im Gegensatz zum fachlichen Modell liegt der Fokus bei der Erstellung des Systemmodells in einer formalen, möglichst vollständigen und konsistenten Beschreibung der Geschäftsprozesse. Hierzu findet häufig die UML Verwendung. Für die Fachabteilungen ist das entstandene Systemmodell nicht mehr uneingeschränkt lesbar. Alle Anforderungen der Fachabteilung müssen modelliert werden, denn nur dann werden sie bei der späteren Implementierung auch berücksichtigt. Neben dem eigentlichen Ablauf des Prozesses müssen auch alle darüber hinaus benötigten Informationen formal modelliert werden. Dies gilt zum Beispiel für die im Prozessablauf benötigten Daten oder die für die einzelnen Prozessschritte verantwortlichen Bearbeiter. Das Systemmodell dient dazu, die Beziehungen zwischen den fachlichen Anforderungen und der entsprechenden IT-Implementierung nachvollziehen zu können. Dies führt dazu, dass Geschäftsprozesse leichter anpassbar sind. Voraussetzung ist jedoch, dass das fachliche Modell und die IT-Umsetzung konsistent gehalten werden.

Das ausführbare Modell wird von IT-Implementierern erstellt. Da im Systemmodell bereits alle fachlich relevanten Informationen modelliert wurden, müssen keine fachlichen Entscheidungen mehr getroffen werden. Die Struktur und die Inhalte des Systemmodells können 1:1 übernommen und so implementiert werden, da dieses bereits formal und vollständig ist. Das ausführbare Modell wird für eine spezielle Zielplattform entwickelt. Die Verantwortung liegt bei der IT, da die Fachbereiche nicht über die nötigen Kompetenzen verfügen. Rückfragen an die Fachabteilung sind nicht mehr erforderlich

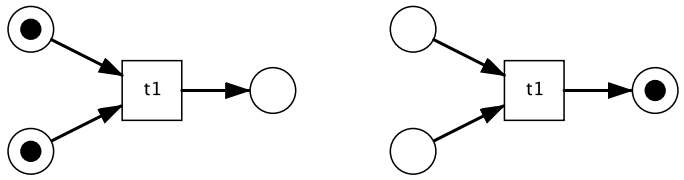
2.2 Modellierungssprachen

Um einen Prozess zu beschreiben kann neben der natürlichen Sprache auch eine grafische Notation verwendet werden. Da ein Prozess im Wesentlichen eine Sequenz von Aktivitäten ist, liegt der Schwerpunkt der Notationen im Modellieren von Abläufen. Darüber hinaus wird teilweise die Möglichkeit geboten, Informationen aus dem Prozesskontext, wie zum Beispiel beteiligte Organisationseinheiten oder Daten, zu modellieren. Im Folgenden werden einige Beispiele von Notationen vorgestellt, die in der Praxis häufig Verwendung finden. Die in dieser Arbeit speziell betrachtete BPMN 2.0 wird in Kapitel 3 detailliert vorgestellt.

2.2.1 Petri-Netze

Die Petri-Netze wurden durch Carl Adam Petri definiert und erstmals 1962 veröffentlicht [Pet62]. Ein Petri-Netz besitzt trotz der wenigen verwendeten Elemente die nötige Grundlage, um Geschäftsprozesse grafisch modellieren zu können [CS94]. Zudem bietet es aufgrund der exakten mathematischen Definition seiner Ausführungssemantik, eine sehr gute Basis für Prozessanalysen. Dies ist ein großer Vorteil gegenüber anderen Modellierungssprachen, bei denen eine Korrektheitsprüfung oft nicht oder nur eingeschränkt möglich ist.

Ein Petri-Netz ist ein bipartiter, gerichteter Graph (siehe Abb. 2.3). Er besteht aus Stellen und Transitionen. Diese werden durch gerichtete Kanten miteinander verbunden. Eine Kante kann nur eine Stelle mit einer Transition verbinden oder umgekehrt. Zwischen zwei Stellen oder zwei Transitionen kann keine Verbindung gezogen werden.



(a) Vor dem Schalten von t1 (b) Nach dem Schalten von t1

Abbildung 2.3: Schaltung eines Petri-Netzes

Stellen werden als Kreise modelliert, Transitionen als Rechtecke. Jede Stelle hat eine Kapazität, die angibt, wie viele Marken enthalten sein können. Durch die Belegung der Stellen mit Marken wird der aktuelle Zustand des Petri-Netzes dargestellt. Jeder Kante ist ein Gewicht zugeordnet, mit dem angegeben wird, was das Aktivieren dieser Kante kostet. Wird bei einer Stelle keine Kapazität oder bei einer Kante kein Gewicht angegeben, wird standardmäßig der Wert eins verwendet.

Eine Transition ist genau dann schaltbar, wenn sich in allen ihr vorgelagerten Stellen mindestens so viele Marken befinden, wie die jeweilige Transition Kosten verursacht. Zudem muss in allen nachfolgenden Stellen genug Kapazität frei sein, um die neuen Marken aufnehmen zu können. Ist dies der Fall, kann die Transition schalten. Dies führt dazu, dass aus den vorgelagerten

Stellen die Marken entsprechend den Gewichten der Transitionen entfernt werden und in den nachfolgenden Stellen Marken entsprechend den Gewichten hinzugefügt werden. Abbildung 2.3a zeigt den Zustand eines Petri-Netzes. Alle Bedingungen zum Schalten der Transition t1 sind erfüllt. Der Zustand nach dem Schalten ist in Abbildung 2.3b dargestellt. Abbildung 2.4 zeigt das Beispiel eines als Petri-Netz modellierten einfachen Geschäftsprozesses. Ein Antrag geht ein. Dieser wird geprüft und liegt nun in einem geprüften Zustand vor. Anschließend wird er entweder abgelehnt oder angenommen. Wie das Beispiel zeigt, werden Stellen dazu benutzt, einen Zustand zu beschreiben. Transitionen hingegen dienen dazu eine Tätigkeit zu beschreiben.

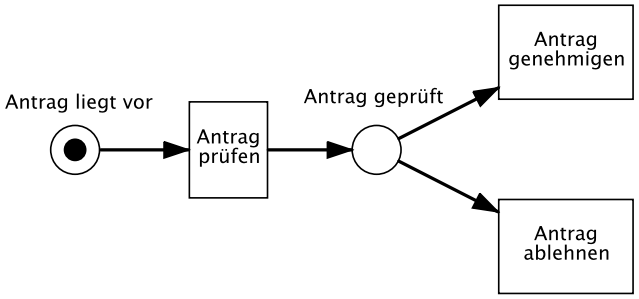


Abbildung 2.4: Beispiel eines Geschäftsprozesses als Petri-Netz

Mit dieser einfachen Form der Petri-Netze können nur einfache Modelle erstellt werden. Um komplexere Modelle erstellen zu können, wurden im Laufe der Zeit mächtigere Petri-Netze definiert. So wurden beispielsweise Transitionen eingeführt, die beim Schalten Zeit verbrauchen [Aal94]. In einer anderen Variante wurde das Petri-Netz um Attribute erweitert, um so auch weitere Informationen modellieren zu können [Jen87]. Mit dieser Erweiterung ist es zum Besipiel auch möglich Daten durch das Petri-Netz zu transportieren.

2.2.2 EPK

Die Ereignisgesteuerte Prozesskette (EPK) wurde 1992 an der Universität des Saarlandes durch eine Arbeitsgruppe unter der Leitung von August-Wilhelm Scheer entwickelt [KNS92]. Die Entwicklung wurde durch die SAP AG angestoßen und wird im Rahmen des ARIS-Konzepts zur Modellierung von Geschäftsprozessen eingesetzt. Auf Basis dieses Konzepts entwickelte die IDS Scheer AG eine BPM-Software, das ARIS Toolset. Das ARIS-Toolset wurde eng mit den ERP Lösungen von SAP gekoppelt. Durch deren weite Verbreitung verbreitete sich auch die EPK und war lange die am häufigsten verwendete Notation zur Prozessmodellierung. Abbildung 2.5 zeigt ein Beispiel einer einfachen EPK. Ein Antrag geht ein und wird anschließend von der Personalabteilung geprüft. Nach der Prüfung wird er entweder abgelehnt oder angenommen. Im Falle einer Ablehnung kümmert sich das Sekretariat um eine Absage. Andernfalls wird er vom Vorgesetzten unterschrieben.

Eine EPK ist ein gerichteter Graph. Er besteht aus aktiven Elementen (Funktionen), passiven Elementen (Ereignissen) und Konnektoren. Ereignisse bilden Zustände ab, in denen sich der

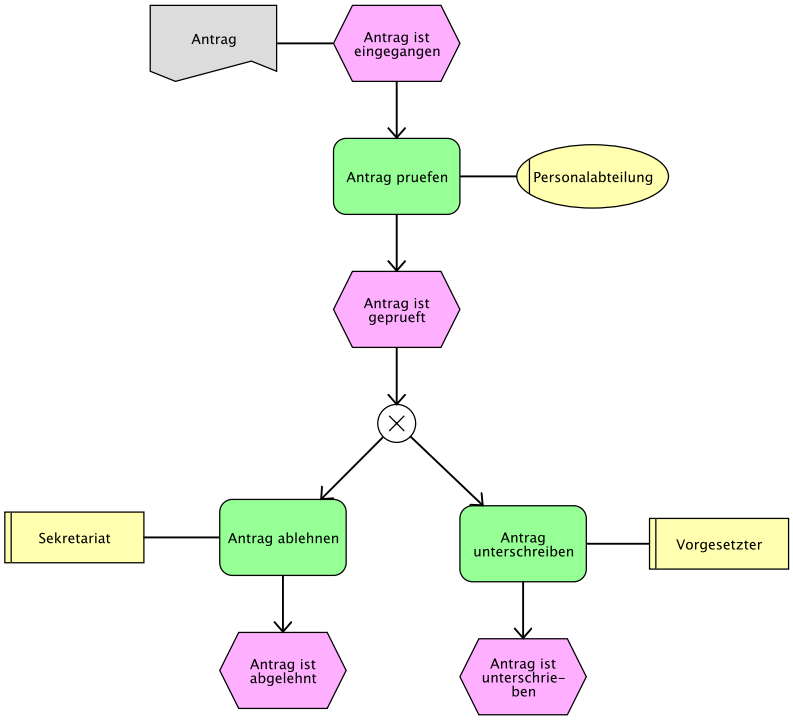


Abbildung 2.5: Beispiel einer EPK

Prozess aktuell befindet. Jeder Prozess muss mit mindestens einem Startereignis beginnen und mit mindestens einem Endereignis enden. Funktionen beschreiben die zu erledigenden Aufgaben. Jede Aufgabe muss auf ein Ereignis folgen. Um den Kontrollfluss zu spalten und zusammenzuführen, existieren logische Konnektoren (UND, ODER, exklusives ODER). Prozesswegweiser verweisen auf andere Prozesse und ermöglichen so das Verknüpfen mehrerer EPKs (nicht im Beispiel dargestellt). Zusätzlich gibt es noch weitere Elemente, um die in einem Prozess beteiligten Organisationseinheiten und Informationsobjekte zu modellieren.

Um auf einem hohen logischen Niveau die Kerngeschäftsprozesse zu definieren und miteinander in Beziehung zu bringen, werden häufig Wertschöpfungskettendiagramme (WKD) verwendet. WKDs fassen Informationen grob auf einem hohen Abstraktionsniveau zusammen. Die in einem WKD definierten Funktionen werden durch EPKs weiter detailliert.

EPKs eignen sich sehr gut dazu, die Abläufe einfach und übersichtlich darzustellen. Beim Modellieren komplexer Sachverhalte stößt man jedoch schnell an Grenzen. Insbesondere wenn es darum geht, so detailiert und präzise zu modellieren, dass das Modell ausgeführt werden kann, ist eine EPK nicht ratsam.

2.2.3 UML Aktivitätsdiagramme

Die UML, ein Standard der OMG, hat ein Repertoire von 13 verschiedenen Diagrammtypen [OMG09c]. Eines davon ist das Aktivitätsdiagramm, mit dem der Ablauf eines Prozesses modelliert werden kann. Es zählt innerhalb der UML zur Gruppe der Verhaltensdiagramme.

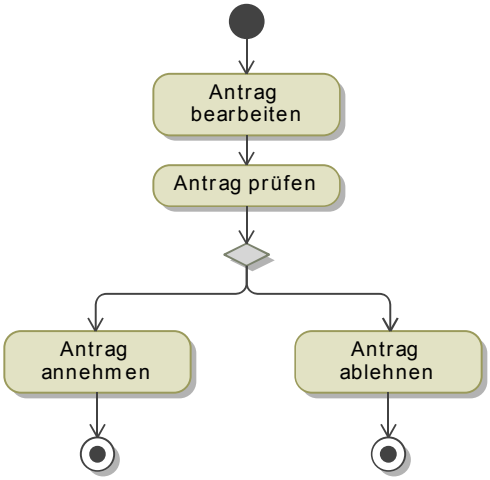


Abbildung 2.6: Beispiel eines UML Aktivitätsdiagramms

Jedes Aktivitätsdiagramm hat einen Start- und einen Endknoten. Dazwischen können Aktivitäten zur Beschreibung von Prozessschritten platziert werden. Der Kontrollfluss wird durch Pfeile dargestellt. Um Verzweigungen darzustellen wird ein Diamantsymbol benutzt, dessen ausgehende Kanten mit Bedingungen beschriftet werden können. Durch Balken kann der Kontrollfluss parallelisiert und später wieder synchronisiert werden. Eine Zuordnung in einzelne Bereiche kann durch Swimlanes vorgenommen werden. Abbildung 2.6 zeigt ein einfaches Aktivitätsdiagramm. Ein Antrag wird bearbeitet, anschließend geprüft und entweder angenommen oder abgelehnt.

Durch UML Aktivitätsdiagramme kann ein Prozess formal und präzise beschrieben werden. Die Notation eignet sich folglich, um in der IT-Abteilung ein ausführbares Prozessmodell zu erstellen. Für eine Fachabteilung sind die entstehenden Modelle aufgrund der technischen Komplexität jedoch kaum verständlich. Ebenso schwierig gestaltet sich für die Fachabteilung das Modellieren einfacher Prozesse mit Aktivitätsdiagrammen, da die Notation aufgrund der Vielzahl an Elementen nicht sehr intuitiv ist und eine längere Einarbeitungszeit benötigt wird.

3

Einführung in die BPMN

Um Geschäftsprozesse zu beschreiben und zu dokumentieren, gibt es verschiedene Möglichkeiten. Die naheliegendste ist, die Prozesse textuell oder tabellarisch zu beschreiben. Um den Ablauf deutlicher zu machen, werden auch oft Grafikprogramme benutzt und mit Kästchen und Pfeilen Ablaufdiagramme gezeichnet. Will man komplexe Prozesse mit allen relevanten Informationen, wie zum Beispiel Verzweigungen, Ereignissen, Daten oder Organisationseinheiten, modellieren, reicht dies jedoch nicht aus. Es wird eine geeignete, standardisierte Modellierungssprache benötigt. Diese muss festlegen, mit welchen grafischen Symbolen welche Sachverhalte modelliert werden, was sie bedeuten und wie sie kombiniert werden können. Eine einheitliche Modellierungssprache ermöglicht es zudem jedem, der die Modellierungssprache beherrscht, von anderen erstellte Modelle zu verstehen. Eine Standardisierung ermöglicht über die reine Dokumentation des Prozesses hinaus auch eine systematische Analyse und Simulation. Die standardisierten Modelle können auch als Grundlage für die Entwicklung eines Informationssystems, das die Abläufe des Geschäftsprozesses unterstützt, verwendet werden.

Es existiert eine Vielzahl an Modellierungssprachen (vgl. Abschnitt 2.2). Von diesen eignet sich jedoch keine, um die Modellierung sowohl auf fachlicher Ebene als auch auf Systemebene durchzuführen. Sie sind entweder nicht formal genug, um ausführbare Modelle zu erstellen, oder zu kompliziert für die Fachabteilung.

Mit der BPMN (Business Process Model and Notation) gibt es einen Standard, der sich durchzusetzen scheint und für sich beansprucht, einfach und gleichzeitig formal zu sein. Die offizielle Webseite `http://www.bpmn.org` listet derzeit 62 BPMN-Implementierungen, vier weitere sind in Planung (Stand Mai 2010). Die Zahl der Veröffentlichungen zum Thema BPMN ist ebenso am Steigen wie die Anzahl an Blogs, die sich mit diesem Thema beschäftigen. Auch in Deutschland erfreut sich die Notation an immer größer werdender Beliebtheit, was Umfragen zeigen [ALF08].

In diesem Kapitel werden die Grundlagen der BPMN erläutert. Dabei wird in Abschnitt 3.1 zuerst auf die historische Entwicklung eingegangen und die Ziele, die bei der Entwicklung verfolgt wurden, werden erläutert. Anschließend werden in Abschnitt 3.2 die Notationselemente der BPMN erläutert. Der letzte Abschnitt dieses Kapitels, 3.3 stellt die verschiedenen Diagrammarten der BPMN vor.

3.1 Entwicklung und Ziele

Die BPMN wurde von der Business Process Management Initiative (BPMI) entwickelt, einem Konsortium, das zum Großteil aus Vertretern von Software-Unternehmen bestand. Der Grund für die Entwicklung war, dass für die Business Process Modeling Language (BPML) [Ark02] eine grafische Notation fehlte. Diese sollte mit der BPMN bereitgestellt werden. Die BPML diente dazu Prozessbeschreibungen zu spezifizieren, die mit einem Business Process Management System ausgeführt werden können. Diese Aufgabe wurde jedoch mehr und mehr von der Business Process Execution Language (BPEL) [OAS07] übernommen, weshalb die Entwicklung der BPML nicht mehr fortgeführt wurde. Doch auch BPEL mangelt es an einer grafischen Notation. Um die Lücke der fehlenden Visualisierung zu schließen, wurde wiederum BPMN verwendet [Sch08, Whi05].

Im Mai 2004 wurde die erste Spezifikation der BPMN veröffentlicht. Im Juni 2005 fusionierte die BPMI mit der Object Management Group (OMG), so dass die BPMN ein Standard der OMG zur grafischen Modellierung von Prozessen wurde. Im Februar 2008 wurde die Version 1.1 veröffentlicht. Zu beachten ist dabei, dass sich die Darstellung der Elemente in einigen Bereichen verändert hat [DS08]. Die letzte offizielle Version trägt die Nummer 1.2, entspricht bis auf einige wenige Korrekturen und Klarstellungen jedoch der Version 1.1 [OMG09a].

Mit der Version 2.0, die voraussichtlich Anfang des zweiten Halbjahres 2010 erscheint, sollen etliche Mängel beseitigt werden. So soll zum Beispiel ein standardisiertes Austauschformat enthalten sein. Es liegt bereits ein offizieller Entwurf vor, der bis zu seiner endgültigen Verabschiedung auch kaum mehr Änderungen erfahren wird [OMG09b]. Schon jetzt arbeiten viele Hersteller von Modellierungswerkzeugen daran, die neue Spezifikation in ihre Programme einzuarbeiten [MID10b, IDS10a, IBM10]. Zudem sind schon einige Bücher erschienen, die die neue Spezifikation erläutern [All09, Sil09, FRH10].

In der Version 2.0 sind die Notationselemente im Vergleich zur Version 1.2 unverändert geblieben, die Anzahl wurde lediglich erweitert. Dies garantiert zum einen, dass die Modelle aufwärtskompatibel sind, und zum anderen, dass Modellierer keine komplett neue Notation erlernen müssen. Hinzugekommen sind zwei neue Diagrammtypen, das Choreographidiagramm und das Konversationsdiagramm. Beide dienen dazu die Interaktion zwischen Geschäftspartnern, die auch bisher schon in Teilen modelliert werden konnte, noch besser darstellen zu können.

Änderungen gab es auch bei den nicht sichtbaren Elementen. Bis zur Version 1.2 lag der Schwerpunkt von BPMN auf der Notation, wohingegen die Semantik nur wenig ausgeprägt war. Eine direkte Ausführung in einer Process-Engine war nicht möglich [Gro07]. Mit der Version 2.0 bekommt die BPMN nun eine formale Definition in Form eines Metamodells. In diesem Metamodell sind neben den im Modell sichtbaren Elementen auch viele Konstrukte beschrieben,

die nicht sichtbar sind. Diese Zusatzinformationen dienen einer formal korrekten Beschreibung mit dem Ziel den Prozess später mit einer Process-Engine auszuführen. Ein großer Mangel war bisher auch das Fehlen eines Austauschformats, was die Wiederverwendung eines Modells in einem anderen Tool erschwerte. Ein Ansatz, der sich jedoch nicht durchgesetzt hat, war der Austausch über XPDL [WFM02, JKJ+04]. Basierend auf dem Metamodell bietet die BPMN nun ein standardisiertes Austauschformat.

Neben der Möglichkeit, Informationen anzugeben, die für eine Ausführung des Prozesses benötigt werden, ist in der neuen Spezifikation auch darauf geachtet worden, den bisher großen Interpretationsspielraum zu beseitigen. Der Schritt in ein ausführbares Prozessmodell war bisher nur durch eine Überführung in BPEL möglich. Die in der Spezifikation angebotene Überführung war jedoch nicht sehr ausgereift. Mit der neuen Spezifikation erhält die BPMN eine eigene Ausführungssemantik, die ein direktes Ausführen des Prozesses ohne eine vorherige Überführung in eine Ausführungssprache ermöglicht. Auf Regeln zur Überführung der BPMN-Modelle in BPEL wurde jedoch nicht verzichtet, so dass nun zwei Möglichkeiten existieren, um einen mit BPMN modellierten Prozess auszuführen. Die Werkzeughersteller können sich selbst entscheiden, welche Variante sie bevorzugen.

Die Abkürzung BPMN stand bisher für "Business Process Modeling Notation". Mit der Version 2.0 und der Einführung eines Metamodels wird auch die Bezeichnung geändert. Der neue Name "Business Process Model and Notation" macht deutlich, dass es sich nun nicht mehr nur um eine reine Notation handelt. Die Abkürzung BPMN bleibt jedoch erhalten.

Das primäre Ziel bei der Entwicklung der BPMN war es, eine einfache Notation zu schaffen. Sie soll für alle in einem Geschäftsprozess beteiligten Personen, vom Implementierer des Geschäftsprozesses bis hin zur entscheidenden Managementebene, leicht verständlich sein. Durch die Einfachheit der Notation und mit Hilfe der modellierten Geschäftsprozesse bietet BPMN die Möglichkeit, innerhalb eines Unternehmens über die Verantwortlichkeitsstufen hinweg einfach zu kommunizieren.

Dabei konzentriert sich die BPMN auf die Modellierung von Prozessen als eine zeitlich-logische Abfolge von Aktivitäten. Folglich kann die BPMN Strukturen wie Aufbauorganisationen, Daten oder Geschäftsregeln nicht abbilden. Manche bemängeln dies und hätten gerne einen Standard, der wie die ARIS-Methodik alle bei der Prozessdokumentation notwendigen Aspekte berücksichtigt [Ste09]. Dies liegt jedoch nicht im Fokus der Entwickler der BPMN und würde auch dem postulierten Prinzip der Einfachheit widersprechen. Für alle im Prozesskontext relevanten Aspekte gibt es bereits Notationen, die teilweise standardisiert sind (z.B. die UML Klassendiagramme für die Datensicht). Die BPMN kann mit anderen Notationen kombiniert werden bzw. bietet Erweiterungspunkte, um so zusätzliche Aspekte abzudecken.

3.2 Notationselemente

Die BPMN besteht aus einer Reihe von grafischen Elementen. Um die Notation der einzelnen Elemente möglichst einfach erfassbar zu machen, wurde zum einen auf die gute Unterscheidungsmöglichkeit dieser geachtet. Zum anderen wurden verwandte Elemente durch eine ähnliche Notation in Bezug zueinander gesetzt. Ein Diagramm kann immer so komplex wie benötigt

modelliert werden. Es wird jeweils ein Ausschnitt der gesamten Elemente verwendet, um den gewünschten Detaillierungsgrad zu erreichen. Dies kann von Anwendungsgebiet zu Anwendungsgebiet verschieden sein. Ein Fachmodellierer erstellt ein weniger detailiertes Modell, bei dem er nur wenige unterschiedliche Elemente verwendet. Ein Mitarbeiter der IT-Abteilung dagegen, der den modellierten Prozess ausführen möchte, wird auf viele verschiedene Elemente zurückgreifen [MR08].

Die Notationselemente sind eingeteilt in Basiskonzepte und erweiterte Konzepte. Die Basiskonzepte beinhalten die wesentlichen Konzepte und reichen aus, ein einfaches Diagramm verständlich zu modellieren. Sollen komplexere Zusammenhänge modelliert werden, kann auf die erweiterten Konzepte zurückgegriffen werden. Im Abschnitt 3.2.1 werden die Basiskonzepte erläutert. Anschließend werden in Abschnitt 3.2.2 die erweiterten Konzepte vorgestellt.

3.2.1 Basiskonzepte

Die Basiskonzepte können in fünf Kategorien eingeteilt werden: `Flow Objects`, `Connecting Objects`, `Data`, `Swimlanes` und `Artifacts`. Sie ermöglichen es, einen Geschäftsprozess vollständig zu beschreiben. Werden lediglich die Basiskonzepte verwendet, ist das Diagramm gut lesbar und leicht verständlich.

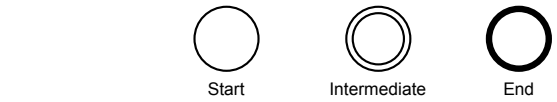
Flow Objects

`Flow Objects` dienen dazu, das Verhalten eines Geschäftsprozesses grafisch zu modellieren. Es existieren drei verschiedene Typen:

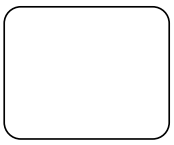
`Events` modellieren Ereignisse, die in Prozessen oder Choreographien auftreten. Die Ereignisse haben entweder eine Ursache (`trigger`) oder eine Wirkung (`result`). Beispiele hierfür sind das Eintreffen einer Nachricht und das Versenden einer Nachricht. Dargestellt wird ein `Event` durch einen Kreis, in dessen Mitte ein Symbol für die Art des `Events` platziert wird. Klassifiziert werden `Events` nach dem Auftreten innerhalb des Prozesses: `Start-`, `Intermediate-` und `End-Event` (siehe Abb. 3.1a).

Eine `Activity` wird als Rechteck mit abgerundeten Ecken dargestellt (siehe Abb. 3.1b). Sie beschreibt eine Aufgabe, die innerhalb eines Geschäftsprozesses zu erledigen ist, wie zum Beispiel "Reservierung bearbeiten". `Activities` können atomar oder nicht atomar sein. Eine atomare `Activity` wird als `Task` bezeichnet. Eine `Activity` kann durch einen eigenen Prozess genauer spezifiziert werden, man spricht von einem `Sub-Process`. Dargestellt wird dieser durch ein kleines Pluszeichen unten in der Mitte des Symbols.

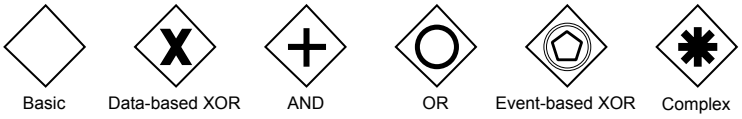
Ein *Gateway* wird als Raute dargestellt und wird dazu benutzt, Kontrollflüsse zu trennen beziehungsweise zusammenzuführen (siehe Abb. 3.1c). Die Art des `Gateways` wird wiederum durch ein Symbol im Inneren der Raute veranschaulicht.



(a) Event



(b) Activity



(c) Gateway

Abbildung 3.1: Flow Objects

Data

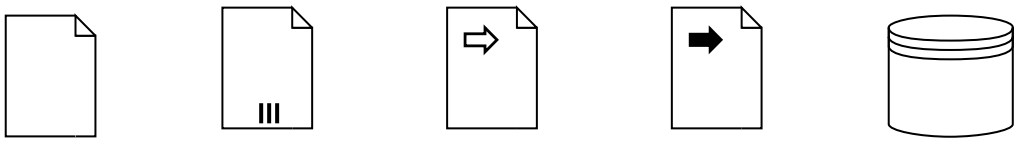
`Data` spezifiziert die Daten, die während der Ausführung eines Geschäftsprozesses als Eingabe benötigt werden und als Ausgabe geschrieben werden. Die einzelnen Elemente werden durch `Associations` mit `Flow Objects` oder einem `Sequence Flow` verknüpft. Wobei eine `Association` die Verbindung mit einem anderen Element beschreibt und ein `Sequence Flow` den Kontrollfluss innerhalb des Prozesses modelliert.

Ein `Data Object` stellt zum einen Informationen bereit, die eine `Activity` für ihre Ausführung benötigt. Zum anderen werden durch `Data Objects` Informationen gespeichert, die von einer `Activity` während ihrer Ausführung erzeugt werden. Ein `Data Object` kann dabei ein einzelnes Objekt repräsentieren (siehe Abb. 3.2a) oder eine ganze Sammlung von Objekten, eine sogenannte `Collection` (siehe Abb. 3.2b). Die Lebensdauer beschränkt sich auf die Laufzeit der Prozessinstanz.

Mit `Data Inputs` und `Data Outputs` können die Ein- und Ausgabedaten eines Prozesses definiert werden. Die Daten können von einzelnen `Activities` gelesen und geschrieben werden. Die `Data Inputs` werden durch einen nicht ausgefüllten Pfeil (siehe Abb. 3.2c), die `Data Outputs` durch einen ausgefüllten Pfeil (siehe Abb. 3.2d) dargestellt.

`Data Stores` stellen einen Mechanismus bereit, um Daten, die über die Laufzeit des Prozesses hinaus existieren, zu lesen und zu schreiben (siehe Abb. 3.2e).

`Properties` haben im Gegensatz zu den anderen Elementen keine grafische Repräsentation und sind im Diagramm nicht sichtbar. `Properties` können `Processes`, `Activities` und `Events` zugeordnet werden und sind nur lokal für diese zugreifbar.



(a) Data Object (b) Collection (c) Data Input (d) Data Output (e) Data Store

Abbildung 3.2: Data

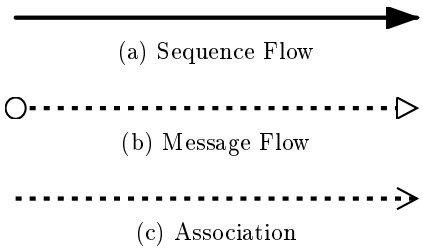
Connecting Objects

`Connection Objects` sind die verbindenden Kanten zur Darstellung des Kontroll- oder Datenflusses sowie der Verbindung von weiteren Elementen innerhalb eines Geschäftsprozesses. Es existieren vier `Connecting Objects`:

Ein `Sequence Flow` verbindet zwei `Flow Objects` und stellt die Ausführungsreihenfolge dar. Dargestellt wird er durch einen Pfeil mit durchgezogenem Strich und ausgefülltem Pfeilende (siehe Abb. 3.3a).

Ein `Message Flow` zeigt den Nachrichtenaustausch zwischen zwei Elementen innerhalb eines Geschäftsprozesses an. Er wird als gestrichelter Pfeil mit einem nicht ausgefüllten Pfeilende dargestellt (siehe Abb. 3.3b). Durch einen `Message Flow` werden immer zwei Teilnehmer des Geschäftsprozesses verbunden. Dargestellt werden diese durch unterschiedliche `Pools`. Die Verknüpfung geschieht entweder direkt oder durch die Verknüpfung mit einem `Flow Object` innerhalb eines `Pools`.

Mit Hilfe einer `Association` können einem `Flow Object Artifacts` assoziiert werden. Dargestellt wird eine `Association` durch eine gepunktete Linie (siehe Abb. 3.3c). Das Pfeilende einer `Association` beschreibt die Flussrichtung.



(a) Sequence Flow

(b) Message Flow

(c) Association

Abbildung 3.3: Connecting Objects

Swimlanes

`Swimlanes` sind Bereiche für Aktoren oder Systeme, innerhalb derer die jeweils zugeordneten Aufgaben dargestellt werden. Es existieren zwei Typen von `Swimlanes`:

Ein `Pool` repräsentiert einen Teilnehmer (ein Benutzer, eine Rolle oder ein System) in einem Geschäftsprozess (siehe Abb. 3.4a). Oft werden `Pools` eingesetzt um Business-to-Business Szenarien zu modellieren.

`Lanes` werden verwendet, um `Pools` weiter zu ordnen und zu klassifizieren. Eine `Lane` unterteilt einen `Pool` in mehrere Bereiche (siehe Abb. 3.4a). Ein Einfügen der `Lanes` ist sowohl vertikal, als auch horizontal möglich.

(a) Pool mit Lanes

(b) Collapsed Pool

Abbildung 3.4: Swimlanes

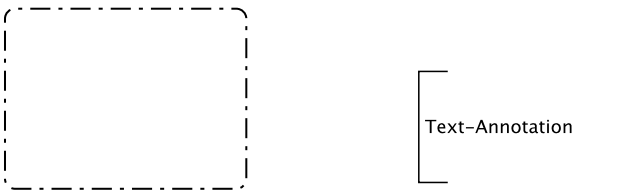
In einem Geschäftsprozess involvierte Teilnehmer können durch `Pools` separiert werden. Innerhalb eines Pools wird jeweils ein eigener Prozess definiert. Dieser muss aber nicht explizit modelliert werden. Ist zum Beispiel der Geschäftsprozess eines Geschäftspartners nicht bekannt, kann dieser in einem `Collapsed Pool` versteckt werden (siehe Abb. 3.4b). Da in jedem Pool ein eigener Prozess modelliert wird, darf ein `Sequence Flow` folglich die Grenzen eines `Pools` nie überschreiten. Um die Kommunikation zwischen zwei `Pools` zu veranschaulichen werden `Message Flows` benutzt.

Artifacts

`Artifacts` ermöglichen es Modellierern weitere Informationen in einem Geschäftsprozess zu beschreiben. Die Struktur des Geschäftsprozesses wird dabei nicht verändert. Die BPMN spezifiziert zwei Typen von `Artifacts`, stellt es dem Modellierer aber frei, weitere Typen selbst zu erstellen:

Eine `Group` gruppiert Elemente. Sie wird durch ein gestricheltes, abgerundetes Rechteck dargestellt (siehe Abb. 3.5a). Eine Gruppierung ist lediglich ein grafisches Element und beeinflusst den Ablauf des Prozesses nicht. Sie wird zur Dokumentation bzw. zur Veranschaulichung von Zusammenhängen zwischen einzelnen Elementen im Geschäftsprozess verwendet.

`Text-Annotations` bieten dem Modellierer die Möglichkeit zusätzliche Informationen für den Betrachter eines Diagramms bereitzustellen (siehe Abb. 3.5b). Sie können durch eine `Association` mit einem beliebigen Element verbunden werden.



(a) Group (b) Text-Annotation

Abbildung 3.5: Artifacts

3.2.2 Erweiterte Konzepte

Um das Modell eines Geschäftsprozesses weiter zu detaillieren, wie es für eine spätere Ausführung nötig ist, gibt es die Möglichkeit, die Elemente durch zusätzliche Variationen und Informationen genauer zu spezifizieren. So können die Geschäftsprozesse formal modelliert werden, ohne dass sich das wesentliche Aussehen des Diagramms ändert. Im Folgenden werden einige, für die weitere Arbeit relevante Konzepte beschrieben.

Activities

`Activities` (vgl. Kapitel 3.2.1) können in atomare und nicht atomare `Activities` unterteilt werden. Eine atomare `Activity` heißt `Task` und kann in verschiedene Typen unterteilt werden. Durch die Angabe eines Typs kann die Art der Bearbeitung genauer definiert werden. Ein `Task`, der nicht weiter spezifiziert wird, wird als `Abstract Task` bezeichnet. Wird der `Task` weiter spezifiziert, ist das an einem Symbol in der linken, oberen Ecke erkennbar (vgl. Abb. 3.6).

Ein `Service Task` ist ein Arbeitsschritt, der einen bereitgestellten Service aufruft, etwa einen Webservice. Dargestellt wird er durch zwei Zahnräder (siehe Abb. 3.6a).

Ein `Send Task` ist ein einfacher Arbeitsschritt, der eine Nachricht an einen externen Partner sendet. Mit einem `Message Flow` kann deutlich gemacht werden, wer der Empfänger der Nachricht ist. Erkennbar sind `Send Tasks` am ausgefüllten Brief-Symbol (siehe Abb 3.6b).

Ein `Receive Task` wartet auf eine eingehende Nachricht eines externen Partners. Ist diese eingegangen, ist der `Task` beendet. Er wird dargestellt durch ein Briefsymbol (siehe Abb. 3.6c) und wird oft dazu verwendet einen Prozess durch einen Nachrichteneingang zu starten.

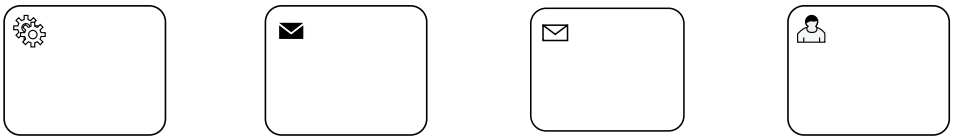
Ein `User Task` ist ein Arbeitsschritt, der das Eingreifen eines Menschen erfordert. Der Bearbeiter wird bei seiner Tätigkeit von einer Software unterstützt. Dargestellt wird dieser `Task` durch ein kleines Männchen (siehe Abb. 3.6d).

Ein `Manual Task` ist ein Arbeitsschritt, der durch einen Menschen ohne Unterstützung eines Computers durchgeführt wird. Ein Beispiel ist das Waschen eines Mietautos, nachdem es zurückgegeben wurde. Das Symbol einer Hand macht deutlich, dass es sich um eine manuelle Tätigkeit handelt (siehe Abb. 3.6e).

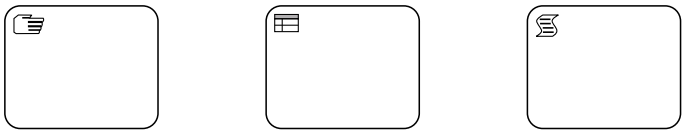
Ein `Business Rule Task` ist eine Schnittstelle zu einer Business-Rule-Engine. Deren Aufgabe ist es, Geschäftsregeln auszuführen und so die Geschäftslogik von der Prozesslogik trennen. Um

diese Aufgabe zu übernehmen, benötigt die Engine Eingabedaten, die im `Business Rule Task` spezifiziert werden können. Gleiches gilt für die Ergebnisse der Berechnung, die von der Engine zurückgeliefert werden. Dargestellt wird ein solcher `Task` durch ein kleines Tabellen-Symbol (siehe Abb. 3.6f).

Ein *Script Task* wird durch die Process-Engine ausgeführt. Der Programmierer schreibt ein Skript, das von der Process-Engine ausgeführt werden kann. Es wird automatisch ausgeführt, sobald alle Eingabebedingungen erfüllt sind. Ist die Ausführung des Scripts beendet, ist auch der `Task` beendet. Abbildung 3.6g zeigt die Darstellung eines `Script Tasks`.



(a) Service (b) Send (c) Receive (d) User



(e) Manual (f) Business Rule (g) Script

Abbildung 3.6: Typen von Tasks

Neben der Zuordnung von Typen können `Tasks` als Loop, Multiple Instance oder Compensation markiert werden. Dies wird durch ein zusätzliches Symbol unten, in der Mitte des `Tasks` deutlich gemacht.

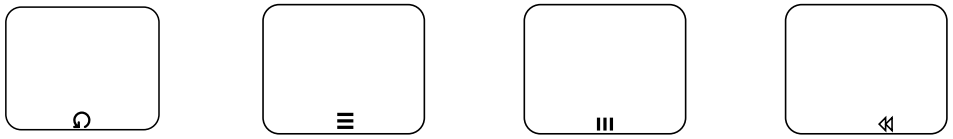
Ein `Loop-Task` (siehe Abb. 3.7a) wird so lange wiederholt, bis eine definierte Bedingung eingetreten ist.

Ein `Multiple-Instance-Task` wird mehrfach instanziiert und kann entweder sequentiell (siehe Abb. 3.7b) oder parallel (siehe Abb. 3.7c) ausgeführt werden.

Ein `Compensation-Task` (siehe Abb. 3.7d) macht Aufgaben, die bereits erledigt wurden, wieder rückgängig und tritt immer in Zusammenhang mit einem `Compensation-Event` auf.

Nicht atomare `Activities` können ebenfalls weiter unterteilt werden. Diese werden als `Sub-Process` bezeichnet und zeichen sich dadurch aus, dass die internen Details weiter modelliert werden. Bei der Verwendung von `Sub-Processes` ist insbesondere darauf zu achten, dass ein `Sequence Flow` im Gegensatz zu einem `Message Flow` und `Associations` nie die Ränder eines Prozesses überschreiten darf.

Ein `Embedded Sub-Process` wird durch ein Plus-Symbol dargestellt (siehe Abb. 3.8a). Dies bedeutet, dass eine weitere Detailierungsstufe aufgerufen werden kann.



(a) Loop (b) MI - Sequential (c) MI - Parallel (d) Compensation

Abbildung 3.7: Weitere Markierungen von Tasks

Mit Hilfe einer `Call Activity` wird auf eine andere `Activity` referenziert. Kommt eine `Activity` an mehreren Stellen vor, muss diese so nur an einer Stelle genauer definiert werden. An den anderen Stellen im Prozess oder auch aus anderen Prozessen kann auf diese verwiesen werden. Grafisch wird eine `Call Activity` durch einen dicken Rand dargestellt (siehe Abb. 3.8b).

Ein `Event Sub-Process` ist ein Teil eines Prozess, der durch das Auftreten eines `Events` ausgelöst wird. Ein `Event Sub-Process` kann unterbrechend oder nicht unterbrechend sein, je nach modelliertem `Event`. Ist er unterbrechend, wird der aktuell laufende Kontrollfluss des Prozesses beendet und lediglich der `Event Sub-Process` ausgeführt. Ist er nicht unterbrechend, werden beide Prozesse parallel ausgeführt. Dargestellt wird er durch eine gestrichelte Linie (siehe Abb. 3.8c).

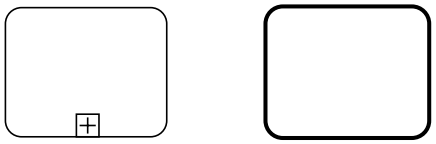
Eine `Transaction` gruppiert `Activities` zu einer logischen Einheit. Für die Gruppe kann ein Transaktionsprotokoll angegeben werden. Besteht der Rand aus zwei Linie, handelt es sich um eine `Transaction` (siehe Abb. 3.8d).

Ein `Ad-Hoc Sub-Process` ist ein Prozess ohne spezifizierte Ablaufreihenfolge. In ihm kann sich eine beliebige Anzahl an `Activities` befinden. Die Reihenfolge und Anzahl der Ausführung wird jedoch erst zur Ausführungszeit bestimmt. Es muss eine Bedingung angegeben werden, wann der `Sub-Process` beendet wird. Erkennbar ist ein `Ad-Hoc Sub-Prozess` an einer Tilde (siehe Abb. 3.8e).

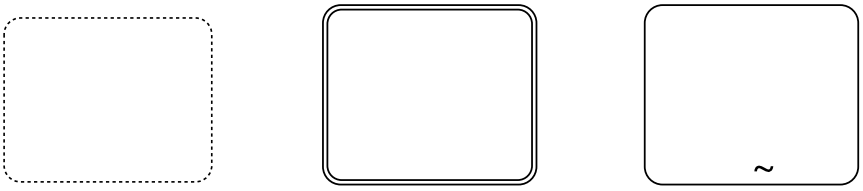
Events

Es existiert eine Vielzahl von `Events` (siehe Anhang A). Neben der bereits erwähnten Aufteilung in `Start-`, `Intermediate-` und `End-Event` kann auch eine Einteilung in `Catching Event` und `Throwing Event` vorgenommen werden.

Ein `Catching Event` wird vom Geschäftsprozess empfangen, ein `Throwing Event` sendet Informationen. Ein `Start-Event` ist folglich stets ein `Catching Event`, da der Prozess, bevor er gestartet wurde, noch keine Informationen zurückgeben kann. Ein `End-Event` ist stets ein `Throwing Event`, da der Prozess sobald er beendet wird, keine Daten mehr empfangen, sondern nur noch senden kann. Dieser Logik folgend, gibt es bei den `Intermediate-Events` sowohl `Catching`, als auch `Throwing Events`. Grafisch können die beiden Typen dadurch unterschieden werden, dass bei einem `Catching Event` das Symbol in der Mitte nicht ausgefüllt ist (siehe Abb. 3.9a), während dieses bei einem `Throwing Event` ausgefüllt ist (siehe Abb. 3.9b).



(a) Embedded (b) Call Activity



(c) Event (d) Transaction (e) Ad-Hoc

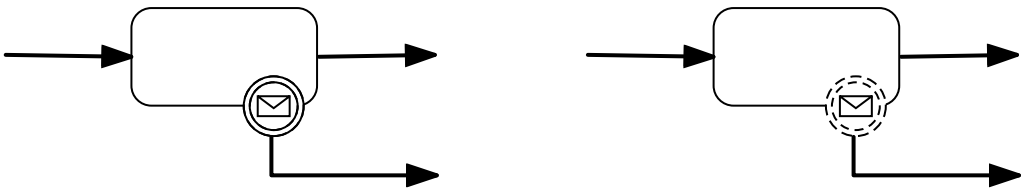
Abbildung 3.8: Sub-Processes

(a) Catching (b) Throwing (c) Interrupting (d) Non-Interrupting

Abbildung 3.9: Weitere Unterteilung der Events am Beispiel des Message-Events

`Events` können bei der BPMN an drei verschiedenen Stellen eingesetzt werden. Am häufigsten finden `Start-`, `Intermediate-` und `End-Events` jedoch im Kontrollfluss Verwendung und beeinflussen dessen Ablauf direkt. `Start-Events` können darüber hinaus in `Event Sub-Processes` als auslösendes `Event` verwendet werden. Dabei gibt es zwei verschiedene Möglichkeiten. Tritt ein `Event` ein, kann entweder ein paralleler Kontrollflusspfad gestartet werden oder der Prozessablauf unterbrochen werden. Dies hätte zur Folge, dass nur mit einem alternativen Pfad fortgefahren wird. Soll das `Event` unterbrechend sein, wird es mit einem durchgezogenen Kreis modelliert (siehe Abb. 3.9c), soll es nicht unterbrechend sein, wird er mit einem gestrichelten Kreis dargestellt (siehe Abb. 3.9d). Zudem können `Intermediate Events` an eine `Activity` angehängt werden, man spricht dann von einem `Boundary Event`. Auch hier gibt es die beiden Möglichkeiten eines `Interrupting Events` (siehe Abb. 3.10a), dass einen alternativen Pfad beginnt und eines `Non-Interrupting Events` 3.10b), dass die `Activity` nicht unterbricht sondern einen zusätzlichen Kontrollflusspfad startet.

Es existiert eine Vielzahl an unterschiedlichen Typen von `Events`, um die verschiedenen Arten von Ereignissen, die während der Ausführung eines Prozesses auftreten können, zu modellieren (siehe Anhang A). Nicht jede Kombination der vorhandenen Typen ist jedoch zulässig. Dies



(a) Interrupting (b) Non-Interrupting

Abbildung 3.10: Boundary Events

schränkt den Modellierer jedoch nicht ein, da lediglich die Konstrukte weggelassen wurden, die innerhalb eines Prozesses keine Daseinsberechtigung haben. Es macht beispielsweise keinen Sinn als `End-Event` einen Timer zu modellieren, weshalb dieses Element in der Spezifikation nicht vorgesehen ist.

3.3 Diagramme

Wie bereits erwähnt, definiert die BPMN 2.0 neben dem Prozessdiagramm und dem Kollaborationsdiagramm zwei neue Diagrammtypen, das Choreographiediagramm und das Konversationsdiagramm [**?**]. Beide dienen der Veranschaulichung der Interaktion zwischen verschiedenen Prozess-Teilnehmern. Da die Version 2.0 noch nicht endgültig verabschiedet ist, existiert noch wenig praktische Erfahrung.

3.3.1 Prozessdiagramm

Ein Prozessdiagramm stellt den Ablauf von Arbeitsschritten innerhalb einer Organisation dar. Die Beschreibung kann so detailliert und präzise sein, dass diese mit Hilfe einer Process-Engine direkt ausgeführt werden kann. Das Diagramm enthält dabei lediglich den Sequenzfluss sowie Datenobjekte und Artefakte, jedoch keine Nachrichtenflüsse. Diese werden im Kollaborationsdiagramm dargestellt. Es enthält lediglich einen `Pool`, der jedoch in mehrere `Lanes` eingeteilt werden kann. Im Vergleich zu den vorigen Versionen der BPMN haben sich die grundlegenden Elemente für die Modellierung eines Prozesses nicht geändert. Um Prozesse noch präziser modellieren zu können, sind einige wenige Elemente neu hinzugekommen. Dies dient dem Ziel die modellierten Geschäftsprozesse direkt ausführbar zu machen.

3.3.2 Kollaborationsdiagramm

Eine Kollaboration stellt das Zusammenspiel mehrerer Partner mittels Nachrichtenaustausch dar. Die beteiligten Partner werden durch `Pools` modelliert. Die `Pools` können als sogenannte

`Black-Box-Pools` gezeichnet werden, ohne den zugehörigen Prozess auszumodellieren. Die Nachrichtenflüsse beginnen dann am Rand des `Pools`. Die `Pools` können aber auch als `White-Box-Pools` modelliert werden. In diesem Fall wird der Prozess in den `Pool` eingezeichnet (mit Hilfe eines Prozessdiagramms). Es kann eine weitere Unterscheidung in öffentliche Prozesse und private Prozesse vorgenommen werden. In öffentlichen Prozessen werden nur die Elemente modelliert, die am Nachrichtenaustausch mit Partnern beteiligt sind. In privaten Prozessen wird der komplette Prozess ausmodelliert.

3.3.3 Choreographiediagramm

In einem Choreographiediagramm wird der Ablauf des Nachrichtenaustauschs zwischen Pools und damit unterschiedlichen Partnern modelliert. Dies ist lediglich eine andere Sicht auf ein Kollaborationsdiagramm. Es wird die genaue Reihenfolge des Austauschens von Nachrichten zwischen den Partnern deutlich gemacht.

Um die Ablaufreihenfolge darzustellen existiert eine `Choreography Activity`, die den Austausch von einer oder mehreren Nachrichten zwischen mehreren Partnern veranschaulicht. Der Sequenzfluss der `Choreography Activities` kann wie in einem Prozess modelliert werden. Einige Elemente, wie zum Beispiel `Message-Events`, sind in diesem Kontext jedoch nicht sinnvoll und daher auch nicht zulässig. Abbildung 3.11 zeigt ein Choreographiediagramm, das grob den Ablauf der Nachrichten modelliert, die zwischen dem Kunden und einer Autovermietung gesendet werden. Der aktive (sendende) Partner ist weiß, der passive (empfangende) Partner ist grau hinterlegt.

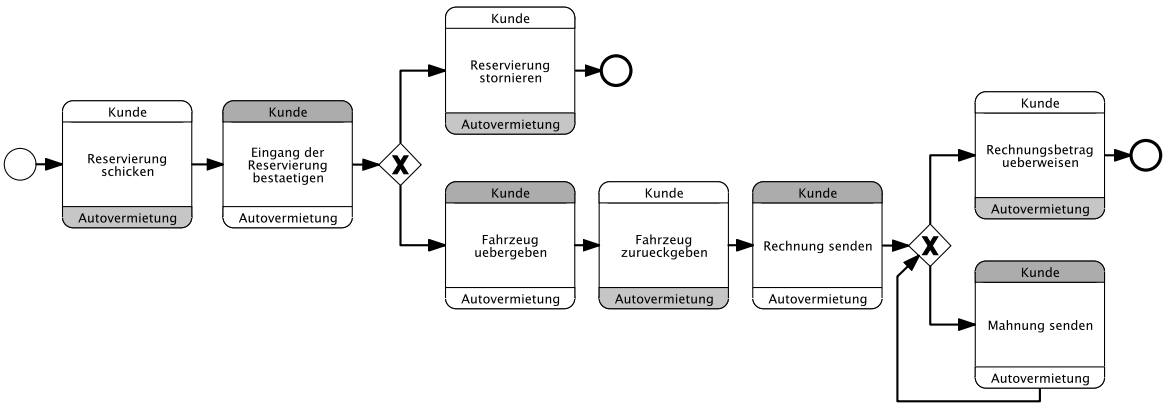


Abbildung 3.11: Beispiel eines Choreographiediagramm

Ein Choreographiediagramm kann in ein Kollaborationsdiagramm eingebettet werden, um den Ablauf des Nachrichtenflusses für den Leser deutlicher zu machen.

3.3.4 Konversationsdiagramm

Mit Hilfe des Konversationsdiagramms können Nachrichtenaustausche zusammengefasst werden. Um eingehende Nachrichten der korrekten Prozessinstanz zuordnen zu können, müssen diese eine eindeutige Kennzeichnung besitzen, den sogenannten `Correlation Key`. Alle Nachrichten mit dem gleichen `Correlation Key` werden in einer `Communication` zusammengefasst. Die `Communications` werden mit sogenannten `Communication Links` mit den an am Nachrichtenaustausch beteiligten `Pools` verbunden. In Abbildung 3.12 ist ein Konversationsdiagramm modelliert, das die Kommunikation zwischen dem Kunden und einer Autovermietungsfirma darstellt. Beide Partner werden als `Pool` modelliert. Die Kommunikation wird durch das Sechseck, die sogenannte `Communication` deutlich gemacht, die durch die `Communication Links` mit den `Pools` verbunden ist.

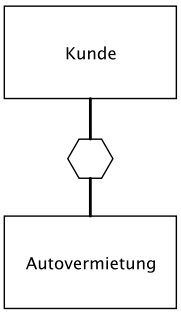


Abbildung 3.12: Beispiel eines Konversationsdiagramms

4

Modellierung auf fachlicher Ebene

Die Geschäftsprozesse eines Unternehmens befinden sich oft nur in den Köpfen der Mitarbeiter. Die Prozesse werden durch die Mitarbeiter gelebt, sind jedoch nicht dokumentiert. Jeder Mitarbeiter kennt einen Ausschnitt des kompletten Prozesses und zwar genau den Teil, in den er involviert ist. Der komplette Prozessablauf kann nur durch die Befragung vieler Mitarbeiter der Fachabteilung ermittelt werden.

Sollen die Prozesse dokumentiert werden, um Kenntnisse über die genauen Abläufe zu bekommen, sie zu optimieren und durch IT zu unterstützen, müssen sie durch die Mitarbeiter aus den Fachabteilungen modelliert werden. Da die beteiligten Mitarbeiter meist keine Kenntnisse in der Geschäftsprozessmodellierung haben, muss eine einfache Möglichkeit geschaffen werden, die in den Köpfen vorhandenen Abläufe zu modellieren. Hierzu ist eine textuelle Beschreibung gut geeignet. Auch einfache grafische Notation, die keine lange Einarbeitungszeit benötigen, finden oft Verwendung.

Die Mitarbeiter aus den Fachabteilungen besitzen zwar keine weitreichenden Kompetenzen in der Geschäftsprozessmodellierung, aber dort sitzt das Wissen über den detaillierten Prozessablauf. Deshalb ist es wichtig, schon im Fachprozess möglichst viele Informationen über die Details des Prozesses zu modellieren. Trotzdem sollte die Arbeit für die Fachmodellierer möglichst einfach sein. Die Devise lautet: So einfach wie möglich, aber trotzdem so präzise wie möglich. Der Aspekt des möglichst frühzeitigen Modellierens wird Frontloading genannt.

Nachdem der grobe Prozessablauf spezifiziert wurde, müssen im Sinne des Frontloadings einzelne Aspekte, wie zum Beispiel die im Prozessablauf relevanten Daten oder die Bearbeiter der einzelnen Schritte, genauer definiert werden. Um diese von einem Fachmodellierer in den Fachprozess zu integrieren, sind generell verschiedene Ansätze denkbar. Diese werden in den folgenden Abschnitten für die einzelnen Aspekte dargestellt. Jeder Ansatz wird bewertet. Dabei wird darauf geachtet, ob der Ansatz für den Fachmodellierer praktikabel ist und wie die Ergebnisse aus dem Fachmodell in das Systemmodell überführt werden können.

Die im vorherigen Abschnitt eingeführte BPMN 2.0 ist eine neue Notation, die zur Modellierung der Geschäftsprozesse verwendet werden kann. Auf den ersten Blick bietet sie eine Vielzahl an Symbolen. Eine Vereinfachung ergibt sich durch die Beschränkung auf eine Untermenge der angebotenen Symbole. Dadurch ist es auch Modellierern aus dem Fachbereich möglich, die BPMN ohne langwierige Schulungen zu nutzen. Für die identifizierten Ansätze zur Modellierung der einzelnen im Geschäftsprozess wichtigen Aspekte wird in den späteren Abschnitten jeweils aufgezeigt, wie diese in BPMN 2.0 realisiert werden können.

Im Folgenden werden einzelne Aspekte detailliert, die der Fachmodellierer bei der Erstellung der Fachprozesse betrachten muss. In Abschnitt 4.1 wird die Bearbeiterzuordnung vorgenommen. Anschließend wird in Abschnitt 4.2 auf die Modellierung von Datenobjekten eingegangen. Abschnitt 4.3 beschäftigt sich mit der Modellierung von Ausnahmen, Abschnitt 4.4 mit Flexibilitätsstellen. Den Abschluss bildet Abschnitt 4.5 mit einer Zusammenfassung.

4.1 Bearbeiterzuordnung

Die Hauptbestandteile eines Prozessdiagramms sind Aktivitäten. Diese beschreiben, was getan wird. Genauso wichtig wie die Frage, was getan wird, ist allerdings auch die Frage, wer etwas tut. Diese Zuordnung muss bei der Modellierung eines Prozessdiagramms vorgenommen werden, um bei einer späteren Ausführung die anfallenden Arbeiten den zuständigen Stellen zuweisen zu können.

4.1.1 Sicht der Fachmodellierer

Die meisten Modellierer des Fachmodells haben wenig IT-Kompetenzen, wissen jedoch, welcher Prozessschritt durch welchen Mitarbeiter bearbeitet wird. Dieses Wissen variiert jedoch stark. Mitarbeiter, die schon lange bei einem Unternehmen beschäftigt sind, wissen beispielsweise mehr, als erst kürzlich eingestellte Mitarbeiter. Alle Informationen, die nicht im Fachmodell modelliert werden, müssen aufwändig durch Rückfragen gewonnen werden. Aus diesem Grund ist es im Zuge des Frontloadings wichtig, den Modellierern des Fachmodells Mechanismen zur Verfügung zu stellen, um die Bearbeiterzuordnung möglichst detailliert vornehmen zu können. Diese Mechanismen müssen so geartet sein, dass ein langjähriger Mitarbeiter mit IT-Kompetenzen sehr detailliert modellieren kann. Gleichzeit muss jedoch auch ein Modellierer ohne große IT-Kompetenzen die Bearbeiterzuordnung modellieren können. Dabei muss er unterstützt werden, um möglichst viele Informationen auf eine für ihn intuitive Art, modellieren zu können.

Neben der oben bereits erwähnten Zuordnung von Bearbeitern zu einzelnen Prozessschritten sind auch noch weitere Verantwortlichkeiten denkbar. Über den Bearbeiter hinaus könnten zum Beispiel Mitarbeiter benannt werden, die die Verantwortung für das korrekte Ablaufen eines Prozessschrittes übernehmen, oder die bei Problemen konsultiert werden können. Im Folgenden wird darauf verzichtet die verschiedenen Gruppen von Verantwortlichen darzustellen. Für eine Betrachtung der Zuordnung genügt es, den Prozessschritten potentielle Bearbeiter zuzuweisen. Die Zuordnung weiterer Verantwortlicher kann in der gleichen Weise durchgeführt werden.

Organisationsstruktur

Innerhalb eines Unternehmens arbeiten oft viele Mitarbeiter, die in einer Organisationsstruktur eingebettet sind. Um eine Bearbeiterzuordnung durchführen zu können, ist es notwendig, die Strukturen eines Unternehmens in einem Organisationsmodell zu beschreiben. Jedes Unternehmen hat eine individuelle Struktur, weshalb keine allgemeingültige Lösung existiert. Abbildung 4.1 zeigt ein auf die Daimler AG zugeschnittenes Metamodell, auf dem die hier vorgestellten Ansätze beruhen. In dem Modell steht der Mitarbeiter im Mittelpunkt. Jeder Mitarbeiter gehört zu einer Organisationseinheit. Jede Organisationseinheit hat einen Manager und kann einer anderen Organisationseinheit unterstellt sein. So lassen sich verschiedene Abteilungen hierarchisch modellieren. Zudem kann ein Mitarbeiter Mitglied einer oder mehrerer Gruppen sein, die jeweils durch einen Manager geleitet wird. Damit können Projektgruppen oder Gremien abgebildet werden. Jeder Mitarbeiter kann darüber hinaus Rollen zugeordnet werden und Kompetenzen besitzen. Über die Rollen können Mitarbeiter in gleiche Aufgabengebiete gruppiert werden (z.B. Testfahrer, Software-Entwickler, etc.). Kompetenzen ordnen den Mitarbeitern weitere Eigenschaften zu, wie etwa das Vorhandensein eines Führerscheins oder die Fähigkeit chinesisch sprechen zu können. Basierend auf einer solchen Organisationsstruktur kann eine Zuordnung der Bearbeiter zu den Prozessschritten durchgeführt werden. Dazu können verschiedene Regeln definiert werden.

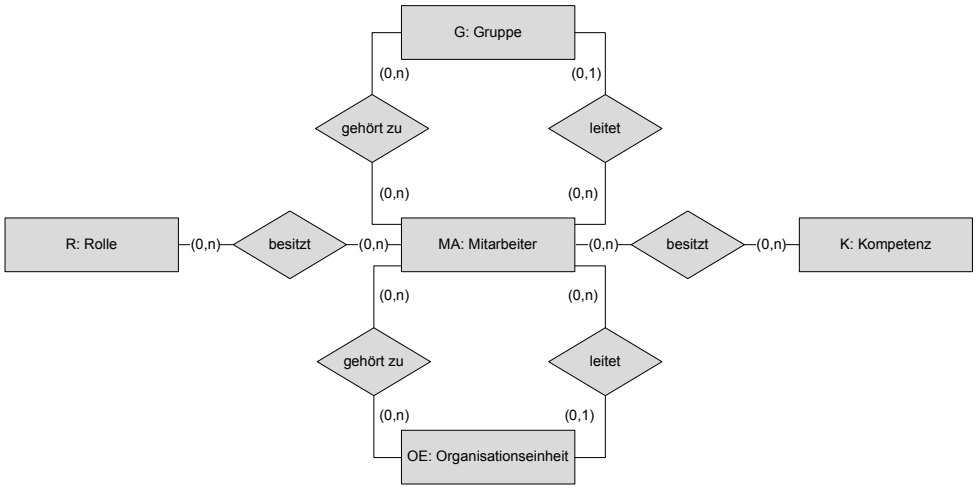


Abbildung 4.1: Metamodell der Organisationsstruktur

Regelset

Bei der Zuordnung von Bearbeitern zu Prozessschritten sind verschiedene Regeln denkbar. Mit einer Regel wird anhand des Organisationsmodells eine Menge an Mitarbeitern, zum Beispiel alle Mitglieder einer bestimmten Organisationseinheit, ausgewählt. Aus dem Organisationsmodell ergibt sich eine Reihe an einfachen Regeln, die auch von einem Modellierer ohne detaillierte Kenntnisse verwendet werden können. Diese werden als Basis-Regeln (siehe Tabelle 4.1) zusammengefasst und beinhalten häufig verwendete Regeln, die einfach zu verstehen sind.

Regel	Bedeutung
MA = x	Bearbeiter ist Mitarbeiter x
R = x	Bearbeiter hat die Rolle x
K = x	Bearbeiter hat die Kompetenz x
G = x	Bearbeiter ist Mitglied der Gruppe x
OE = x	Bearbeiter ist Mitglied der Organisationseinheit x
MA(v) = x	Bearbeiter ist Vorgesetzter des Mitarbeiters x
G(l) = x	Bearbeiter ist Leiter der Gruppe x
OE(l) = x	Bearbeiter ist Leiter der Organisationseinheit x

Tabelle 4.1: Basis-Regeln zur Bearbeiterzuordnung

Mit Hilfe dieser Basis-Regeln lässt sich nun darstellen, dass der Prozessschritt A.1 "Reservierung bearbeiten" des Anwendungsszenarios von einem Mitarbeiter mit der Rolle Sachbearbeiter bearbeitet werden soll:

R = "Sachbearbeiter"

Manche Zuordnungen erfordern die Kombination mehrerer Regeln. Im Falle des Prozessschrittes A.1 "Reservierung bearbeiten" soll der Mitarbeiter nicht nur Sachbearbeiter sein, sondern zusätzlich aus der Abwicklungsabteilung. Die Kombination solcher Regeln kann mit Hilfe von logischen Operatoren realisiert werden. Das eben genannte Beispiel würde folgendermaßen umgesetzt werden:

R = "Sachbearbeiter" `AND` OE = "Abwicklungsabteilung"

Über die Basis-Regeln hinaus gibt es weitere, komplexere Regeln, die Bearbeiter anhand des Organisationsmodells zuordnen. Zum Beispiel alle Mitglieder einer Organisationseinheit auf einer bestimmten Stufe. Diese werden weniger häufig verwendet als die Basis-Regeln und werden deshalb als Komplexe-Regeln (siehe Tabelle 4.2) zusammengefasst. Des Weiteren hängt die Zuordnung von Bearbeitern zu Prozessschritten nicht nur vom Organisationsmodell ab. Auch der Kontext des Prozesses (z.B. vorhandene Daten) und bereits durchgeführte Prozessschritte (z.B. Bearbeiter von Prozessschritt A) müssen beachtet werden. Der Prozessschritt A.4 "Fahrzeug auswählen" soll beispielsweise durch den selben Mitarbeiter durchgeführt werden, welcher bereits Prozessschritt A.1 "Reservierung bearbeiten" bearbeitet hat. Diese Regeln sind komplizierter in der Anwendung und werden ebenfalls als Komplexen-Regeln beschrieben. Mit Hilfe dieser Komplexen-Regeln können versierte Modellierer eine detaillierte Bearbeiterzuordnung vornehmen. Durch das Ausblenden der Komplexen-Regeln kann die Modellierung vereinfacht werden, sodass auch weniger versierte Benutzer die Bearbeiterzuordnung modellieren können, ohne überfordert zu werden.

Durch die Anwendung der Komplexen-Regeln kann die Bearbeiterzuordnung des Prozessschrittes A.4 "Fahrzeug auswählen" realisiert werden.

%FahrzeugAuswählen.bearbeiter%

Zuordnung	Bedeutung
OE(+) = x	Bearbeiter ist Mitglied einer Organisationseinheit, die der Organisationseinheit x übergeordnet ist
OE(-) = x	Bearbeiter ist Mitglied einer Organisationseinheit, die der Organisationseinheit x unterstellt ist
OE(s) = x	Bearbeiter ist Mitglied einer Organisationseinheit der Ebene x
OE(u) = x	Bearbeiter ist Mitglied der Organisationseinheit, die durch x geleitet wird
G(u) = x	Bearbeiter ist Mitglied der Gruppe, die durch x geleitet wird
%x.bearbeiter%	Bearbeiter ist derselbe, der auch schon Prozessschritt x bearbeitet hat
%x.potBearbeiter%	Bearbeiter sind dieselben, die auch schon Prozessschritt x bearbeiten hätten können
$x.y$	Verweis auf Attribut y eines Datums x

Tabelle 4.2: Komplexe-Regeln zur Bearbeiterzuordnung

Zudem können nun auch komplexere Bearbeiterzuordnungen definiert werden. Dabei kann die Auswertungsreihenfolge durch runde Klammern festgelegt werden. Ein Prozessschritt, in dem ein Antrag genehmigt wird, muss beispielsweise von einem Mitglied des Vorstands genehmigt werden. Der Genehmigende darf allerdings nicht der Antragssteller sein. Zudem darf nicht der selbe Mitarbeiter den Antrag genehmigen, der diesen vorher bereits geprüft hat.

OE(s) = “Vorstand” `AND NOT` (MA = $Antrag.Antragssteller$) `AND NOT` (%AntragPruefen.bearbeiter%)

Um die Bearbeiterzuordnung in ein Prozessmodell zu integrieren, muss das zugrunde liegende Organisationsmodell in das Prozessmodell integriert werden. Dies kann auf zwei verschiedene Arten geschehen. Das Organisationsmodell kann entweder direkt in das Prozessmodell integriert werden oder in einem externen Modell beschrieben werden, welches vom Prozessmodell referenziert wird. Wird das Organisationsmodell ins Prozessmodell integriert, wird letzteres stark aufgebläht. Dadurch geht die Übersichtlichkeit verloren, die insbesondere im Fachmodell von großem Vorteil ist. Schon allein aus diesem Grund ist es sinnvoll, das Organisationsmodell auszulagern und lediglich darauf zu referenzieren.

Darstellung der Bearbeiterzuordnung im Prozessmodell

Nachdem das Organisationsmodell definiert wurde, wird den einzelnen Prozessschritten ein Bearbeiter zugeordnet. Hier gibt es verschiedene Ansätze, die im Folgenden detailliert und bewertet werden.

Freie textuelle Beschreibung:
Durch eine Text-Annotation an einem Prozessschritt wird die Bearbeiterzuordnung beschrieben. Die Beschreibung unterliegt keinerlei Vorschriften, sondern ist völlig frei. Das hat den Vorteil, dass auch Modellierer, die keinerlei Kenntnisse von IT haben, die Bearbeiterzuordnung umgangssprachlich beschreiben können. Gleichzeitig ist das Fehlen von festen Vorgaben jedoch auch ein

Nachteil. Modellierer werden dazu verführt, die Bearbeiterzuordnung unsauber zu beschreiben, obwohl sie von ihren Fähigkeiten her in der Lage wären, die Zuordnung formal zu beschreiben. Ein weiterer negativer Punkt ist, dass der entstandene Text manuell in das Systemmodell überführt werden muss.

Templatemechanismus:
Der Modellierer kann die Bearbeiterzuordnung durch die Auswahl von vordefinierten Templates festlegen. Die Templates bestehen aus einem festen Text mit eingebetteten Lücken (Variablen). Diese Lücken müssen durch den Modellierer gefüllt werden (vgl. Abb. 4.2). Die Templates können so gewählt werden, dass mit je einem Template eine der oben erwähnten Regeln abgebildet werden kann. Der Text des Templates sollte sich an dem Erklärungstext der jeweiligen Regel aus Tabelle 4.1 und Tabelle 4.2 orientieren. Ein Template für die Regel "K = x" wäre demnach "Der Bearbeiter hat die Kompetenz X", wobei X die Lücke repräsentiert, die der Modellierer füllen muss. In diesem Fall könnte dies zum Beispiel die Kompetenz "chinesisch" sein. Das ausgefüllte Template würde demnach "Der Bearbeiter hat die Kompetenz *chinesisch*" lauten. Durch einen Abgleich mit dem Organisationsmodell kann sichergestellt werden, dass die Lücken nur mit erlaubten Werten gefüllt werden. Um komplexere Zuordnungen zu realisieren, können die Templates durch logische Operatoren kombiniert werden. Die Templates sind im Grunde nichts anderes, als eine andere Darstellungsform der oben definierten Regeln. Deshalb können Sie mit Hilfe eines einfachen Mappings in die kompakten Regeln überführt und so abgespeichert werden.

Abbildung 4.2 zeigt einen Assistenten, der die Bearbeiterzuordnung mit Hilfe von Templates unterstützt. Um diesen möglichst übersichtlich zu halten, wurden lediglich die Basis-Regeln angegeben und eine Verknüpfung der Templates durch Operatoren ist nicht möglich. Eine Erweiterung auf alle Regeln und eine Verknüpfung durch Operatoren ist jedoch ohne weiteres möglich.

Abbildung 4.2: Bearbeiterzuordnung mit Hilfe von Templates

Durch die natürliche Art der Darstellung als Text, ist es für Modellierer relativ einfach die Bearbeiterzuordnungen durchzuführen. Ein Template mit dem Text "Der Bearbeiter hat die Kompetenz chinesisch" ist für jeden verständlich. Durch die Kombination von Templates mit logischen Operatoren und die Verwendung der erweiterten Regeln wird die Zuordnung komplexer, ist aber trotzdem noch verständlich. Durch die feste Vorgabe der Templates besteht jedoch keine

Flexibilität. Will ein Modellierer eine Zuordnung darstellen, für die es kein Template gibt, kann er dies nicht tun. Da sich die Templates an den Regeln orientieren, ist eine Überführung ins Systemmodell relativ leicht möglich und lässt sich weitestgehend automatisieren.

Modell:
Die Bearbeiterzuordnung kann auch grafisch in einem Diagramm festgelegt werden. Dazu kann das Prozessdiagramm um diese Funktionalität erweitert werden. Alternativ ist es möglich ein weiteres Diagramm hinzuzufügen, auf das von den einzelnen Prozessschritten verwiesen wird. Wird die Bearbeiterzuordnung innerhalb des Prozessdiagramms vorgenommen, ist diese auf zweierlei Arten denkbar. Zum einen ist es möglich, den Prozess in Bereiche aufzuteilen, wobei ein Bereich dann zum Beispiel einer Organisationseinheit entspricht. Alle Prozessschritte innerhalb dieses Bereiches werden von Mitarbeitern dieser Organisationseinheit bearbeitet. Zum anderen kann die Zuordnungen direkt mit den Prozessschritten verbunden werden. Ein Beispiel wird in Abbildung 4.3 gezeigt. Die Art der Notation sollte dabei so gewählt werden, dass sie für den Modellierer einfach verständlich ist, dabei aber möglichst die oben definierten Regeln verwendet. Um die Abhängigkeit zu den Bearbeitern anderer Prozessschritte oder zu Daten kenntlich zu machen, ist es zudem denkbar mit Verbindungslinien zwischen den jeweiligen Elementen zu arbeiten.

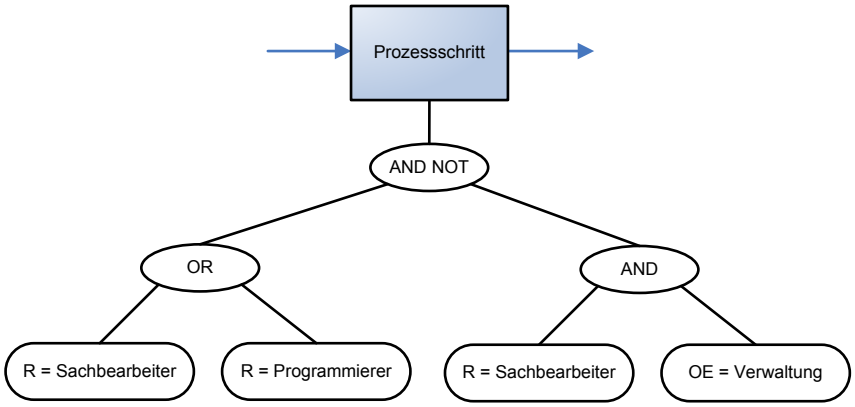


Abbildung 4.3: Grafische Bearbeiterzuordnung mit Hilfe eines Baumes

Wird die Bearbeiterzuordnung innerhalb des Prozessdiagramms modelliert, kann dies schnell sehr unübersichtlich werden. Einfache Zuordnungen können grafisch ansprechend und intuitiv über das Einteilen in Bereiche geschehen. Komplexere Ausdrücke wie in Abbildung 4.3 können so aber nicht mehr sinnvoll dargestellt werden. Durch das Anhängen eines Grafen an jeden Prozessschritt, wird dieses jedoch ebenfalls sehr unübersichtlich. Aus diesem Grunde ist es sinnvoll die grafische Zuordnung in einem externen Diagramm auszulagern. Der Prozessschritt referenziert dann lediglich auf das externe Diagramm, welches die Bearbeiterzuordnung abbildet. Ein weiterer Vorteil dabei ist, dass bestehende Notationen für die Modellierung genutzt werden können. Die Überführung in das Systemmodell kann bei einer Notation, die die oben beschriebenen Regeln grafisch umsetzt, weitgehend automatisch erfolgen.

Text, der formaler Grammatik genügt:
Wie bei der freien textuellen Beschreibung werden Texte verfasst. Diesem liegt nun jedoch eine formale Grammatik zugrunde. Die Aufgabe für den Modellierer wird dadurch erschwert, dass er nun nicht mehr wahllos Text schreiben kann, sondern sich an die gegebene Grammatik halten

muss. Durch die Grammatik ist die Zuordnung jedoch formal und kann leicht in das Systemmodell überführt werden. Das verwendete Tool kann den Modellierer durch eine automatische Syntaxprüfung unterstützen und darüber hinaus sicherstellen, dass nur im Organisationsmodell definierte Begriffe verwendet werden.

Die oben bereits verwendete Notationsform ist eine Möglichkeit die Bearbeiterzuordnung durch einen Text, der einer formalen Grammatik genügt, darzustellen. Im Folgenden wurde das Beispiel aus Abb. 4.3 in dieser Form umgesetzt.

(R = "Sachbearbeiter" `OR` R = "Programmierer") `AND NOT` (R "Sachbearbeiter" `AND` OE = "Verwaltung")

Fazit:
Im Rahmen des Frontloadings sollen schon in einer frühen Phase der Modellierung möglichst viele Informationen gesammelt werden. Um dies zu erreichen, muss dem Modellierer eine möglichst einfache und intuitive Möglichkeit gegeben werden, die nötigen Informationen in das Modell einzupflegen. In diesem Punkt sind insbesondere die Hersteller der entsprechenden Modellierungssoftware gefragt. Um auf die verschiedenen Fähigkeiten der Modellierer Rücksicht zu nehmen, dient die Unterscheidung in Basis-Regeln und Komplexe-Regeln. Um weniger kompetente Benutzer nicht zu überfordern, sollten lediglich die Basis-Regeln bereitgestellt werden, für fachkundige Benutzer sollten darüber hinaus aber auch die Komplexen-Regeln verfügbar sein.

Um eine möglichst formale Darstellung zu haben, die leicht in das Systemmodell überführt werden kann, zugleich jedoch möglichst benutzerfreundlich ist, eignen sich Templates am Besten. Unter Umständen ist es jedoch nicht möglich, alle erdenklichen Fälle mit Hilfe der Templates zu modellieren. Dies ist der Fall, wenn nicht für alle Eventualitäten ein Template verfügbar ist. Denkbar wäre zum Beispiel eine Prozessschritt, der innerhalb einer Schleife modelliert ist und mehrfach ausgeführt werden kann. Die Bearbeiter unterscheiden sich jedoch je nachdem, in welchem Durchlauf sich die Schleife befindet. Außerdem kann es passieren, dass der Benutzer mit der Verwendung der Templates, insbesondere was den Verweis auf frühere Prozessschritte und Daten angeht, überfordert ist. Für diese Fälle ist es sinnvoll, ein zusätzliches Feld für einen Freitext vorzusehen.

4.1.2 Umsetzung in BPMN

Die Bearbeiterzuordnung ist ein Thema, dass in der BPMN zwar Beachtung findet, jedoch nicht auf erschöpfende Art und Weise. So fehlt im Entwurf zur BPMN 2.0 bislang eine Möglichkeit zur Integration eines Organisationsmodells. Es wurden jedoch verschiedene Elemente definiert, um eine Bearbeiterzuordnung durchzuführen. Teilweise werden diese im Diagramm dargestellt, teilweise sind sie lediglich im zugrunde liegenden XML vorhanden.

Mit Hilfe von `Swimlanes` (vgl. Abschnitt 3.2.1) lässt sich der Prozess in verschiedene Bereiche untergliedern. Durch das Vergeben von sprechenden Namen für die einzelnen Bereiche kann deutlich gemacht werden, wer für die Durchführung der `Activities` innerhalb dieser Bereiche verantwortlich ist. In der zugrunde liegenden XML-Datei lässt sich jedoch lediglich ein Name für die einzelnen `Pools` und `Lanes` festlegen. Eine genauere Definition ist nicht möglich. `Lanes` lassen

sich zwar hierarchisieren und eignen sich dazu, die grobe Struktur innerhalb eines Unternehmens beziehungsweise einer Abteilung darzustellen. Eine weitere Unterteilung in Rollen oder Kompetenzen ist jedoch nicht praktikabel. Eine einfache Angabe etwa einer Organisationseinheit reicht für die Bearbeiterzuordnung jedoch oft nicht aus. Mit Hilfe der `Swimlanes` kann folglich keine komplexe, formal korrekte Bearbeiterzuordnung erfolgen. Sie dienen lediglich dazu, eine grobe Zuordnung der `Activities` grafisch deutlich zu machen. Insbesondere für die Lesbarkeit und die grobe Einteilung von Diagrammen eignen sie sich jedoch gut. Dies ist wichtig für die Akzeptanz der modellierten Prozesse innerhalb des Unternehmens.

Über die grafische Zuordnung durch `Swimlanes` hinaus, können einzelnen `Activities` `Performer` zugeordnet werden. Das BPMN-Element `Performer` definiert eine, für die Ausführung einer `Activity` verantwortliche Arbeitskraft. Das Element `Performer` verweist wiederum auf eine `Resource`. Eine `Resource` ist in einem Diagramm nicht sichtbar und repräsentiert ein "Betriebsmittel". Neben menschlichen Arbeitskräften, können dies auch Computersysteme, Drucker, Scanner, ...sein. Eine `Resource` kann von mehreren Stellen referenziert werden, sodass mehreren `Activities` dieselbe `Resource` als `Performer` zugeordnet werden kann.

`Human Tasks` und `Manual Tasks`[1], die eine Interaktion mit menschlichen Benutzern voraussetzen, kann neben dem `Performer` noch ein `HumanPerformer` und ein `PotentialOwner` zugewiesen werden. Dies geschieht wiederum durch eine Referenz auf eine `Resource`. Im Gegensatz zu einem `Performer` ist ein `HumanPerformer` und ein `PotentialOwner` in jedem Fall eine menschliche Arbeitskraft. Ansonsten unterscheiden sie sich jedoch nicht, haben also auch keine weiteren Attribute.

Einer `Resource` können beliebig viele Parameter zugeordnet werden. Die Parameter bestehen aus einem Namen, einem Typ und einem Attribut, das angibt, ob der Parameter gefordert wird oder nicht. Der XML-Code einer `Resource`, die den Leiter einer Niederlassung repräsentiert, könnte zum Beispiel wie in Listing 4.1 aussehen. Durch den Parameter Stadt kann die jeweils passende Niederlassung gefunden werden. Voraussetzung dafür ist, dass in jeder Stadt maximal eine Niederlassung existiert.

```
01 <resource id="niederlassungLeiter" name="Leiter einer Niederlassung">
02   <resourceParameter id="paramStadt" isRequired="true" name="Stadt" ↩
03   type="xsd:string"/>
04 </resource>
```

Listing 4.1: XML-Code einer Resource

Die Definitionen können durch die direkte Eingabe von XML-Code vorgenommen werden. Wie bei allen folgenden XML-Beispielen ist es jedoch sinnvoll, den Modellierer durch eine einfache und strukturierte Eingabemaske zu unterstützen. Zum einen vereinfacht dies die Eingabe für den Modellierer und zum anderen sinkt dadurch die Fehlerrate bei der Modellierung.

Wird in einer `Activity` auf eine `Resource` referenziert, kann eine Zuordnung über die Parameter erfolgen. Dies geschieht im Regelfall anhand der im Prozesskontext vorhandenen Daten. Mittels `Expressions` kann auf diese zugegriffen werden (siehe Kapitel 4.2.2). Durch `Expressions` kann

[1] beides sind Spezialisierungen einer `Activity`

durch eine Anfragesprache (etwa XPath) auf Teile der Daten zugegriffen werden und diese in Verbindung mit den Parametern der `Resourcen` gebracht werden. So kann modelliert werden, dass ein Fahrzeug immer vom Leiter der Niederlassung freigegeben werden muss, in der das Auto abgeholt werden wird (vgl. Listing 4.2).

```
01 <userTask id="fahrzeugFreigeben" name="Fahrzeug freigeben">
02   ...
03   <!-- weitere Definitionen -->
04   ...
05   <potentialOwner resourceRef="tns:niederlassungLeiter">
06     <resourceParameterBinding parameterRef="tns:paramStadt">
07       <formalExpression>getDataInput('auftrag')/startort< ↩
08       /formalExpression>
09     </resourceParameterBinding>
10   </potentialOwner>
11 </userTask>
```

Listing 4.2: XML-Code eines User Tasks, dem eine Resource zugewiesen wird

Von Zeile 5 bis 10 wird für den `User Task` "Fahrzeug freigeben" ein `PotentialOwner` definiert. Dieser wird mit der `Resource` verknüpft, die in Listing 4.1 erstellt wurde (Zeile 5). Der Parameter "Stadt" der `Resource` (Zeile 6) wird mit dem Attribut "Startort" des Datenobjektes "Auftrag" verknüpft (Zeile 7+8).

Mit Hilfe der `Resources` ist eine Bearbeiterzuordnung möglich. Dies reicht jedoch nicht aus, um den komplexen Anforderungen zu genügen. Deshalb werden im Folgenden unterschiedliche Erweiterungen diskutiert, um dennoch eine möglichst vollständige Bearbeiterzuordnung mittels BPMN realisieren zu können.

Um eine Bearbeiterzuordnung durchführen zu können, ist das Vorhandensein eines Organisationsmodells eine Grundvoraussetzung. Wie bereits erwähnt, ist dafür in der BPMN kein eigenes Element vorgesehen. Als Workaround lässt sich mit Hilfe der Resource-Klasse ein Organisationsmodell erstellen. Diese kleine Zweckentfremdung ist so wohl nicht gedacht, jedoch praktikabel. Es bleibt abzuwarten, ob in der endgültigen Spezifikation der BPMN 2.0 eine Möglichkeit besteht, ein Organisationsmodell unabhängig von `Resources` zu modellieren.

Unabhängig von der weiteren Entwicklung der BPMN kann diese durch eigene Erweiterungen um ein Organisationsmodell ergänzt werden. Die Modellierung der Organisationsstruktur kann so direkt im BPMN Diagramm erfolgen. Der Schwerpunkt der BPMN liegt jedoch auf der Modellierung des Prozessablaufs. Darüber hinausgehende Informationen, die im BPMN-Diagramm modelliert werden, machen dieses unübersichtlicher. Da bereits andere Notationen zur Modellierung von Organisationsmodellen existieren, sollte auf diese zurückgegriffen werden. Das Organisationsmodell kann in einem externen Diagramm, wie etwa einem UML Objektdiagramm, modelliert werden. Dieses Diagramm wird mit dem Prozessdiagramm verknüpft, sodass die Informationen dort verfügbar sind. Dies wird in BPMN durch eine `External Relationship` realisiert. Listing 4.3 zeigt eine solche Verknüpfung. Nun kann etwa durch eine Process-Engine die Bearbeiterzuordnung ausgewertet werden.

```
01 <definitions id="example"
02   xsi:schemaLocation="http://www.omg.org/bpmn20 Core-Common.xsd"
03   xmlns="http://www.omg.org/bpmn20"
04   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
05   xmlns:tgt="http://www.example.org/OrgModel">
06
07   <import importType="http://www.omg.org/uml" ↩
08   location="organisationsmodell.xmi" namespace="http://www.example. ↩
09   org/OrgModel"/>
10
11   <relationship type="defOrgModel" id="orgModel" direction="forward">
12     <documentation>Diese Beziehung stellt eine Verlinkung zum ↩
13     Organisationsmodell her</documentation>
14     <target ref="tgt:OrgModel"/>
15   </relationship>
16   ...
17   <!-- weitere Definitionen -->
18   ...
19 </definitions>
```

Listing 4.3: External Referenz auf ein Organisationsmodell

In Zeile 7-9 wird das externe Diagramm importiert und dabei der Typ des Diagramms angegeben. Zeile 11-15 erstellt eine Beziehung des Prozessdiagramms zum importierten Diagramm.

Wie bereits in Abschnitt 4.1.1 vorgestellt, lassen sich Bearbeiterzuordnungen im Fachmodell durch die Verwendung von Templates und eventuell einem zusätzlichen Freitext realisieren. Die Templates bestehen aus Text und können als solcher im Diagramm eingefügt werden. Dazu existieren verschiedene Möglichkeiten. Die einfachste ist es, eine `Text-Annotation` zu erstellen und diese mit einer `Activity` durch eine `Association` zu verbinden. Damit klar wird, dass es sich bei der `Text-Annotation` um eine Bearbeiterzuordnung handelt, ist das Verwenden eines Schlüsselwortes, wie zum Beispiel “Bearbeiter”, am Anfang des Textes sinnvoll. Das Attribut `documentation` bietet eine weitere Möglichkeit einen Text zu speichern, ohne die BPMN erweitern zu müssen. Bei jeder `Activity`, wie auch jedem anderen Element innerhalb der BPMN, können beliebig viele `documentation`-Attribute hinzugefügt werden. Diese bestehen lediglich aus einem String, es kann also beliebiger Text gespeichert werden. Um das Element näher zu beschreiben wurde in Listing 4.3 etwa der `Relationship` solch ein Attribut hinzugefügt (Zeile 12+13). Die Bearbeiterzuordnung kann in einem solchen `documentation`-Attribut gespeichert werden. Auch dabei ist die Verwendung eines Schlüsselwortes sinnvoll.

Sowohl die Beschreibung in `Text-Annotations`, als auch die in `documentations` sind zwar möglich, jedoch nicht sehr sinnvoll. Eine elegantere Lösung besteht darin, die Erweiterungsmechanismen der BPMN zu nutzen. Mit deren Hilfe können die `Activities` um ein weiteres Attribut erweitert werden, das die Bearbeiterzuordnung aufnimmt (siehe Listing 4.4).

Das so definierte neue Attribut kann nun in einer Instanz verwendet werden. In Listing 4.5

```
01 <xsd:schema ...>
02   ...
03   <xsd:element name="staffAssignment" type="xsd:string"/>
04   ...
05 </xsd:schema>
```

Listing 4.4: Definition eines weiteren Attributs

wird mit Hilfe des Attributs dem `User Task` "Reservierung bearbeiten" ein Bearbeiter aus der "Abwicklungsabteilung" und der Rolle "Sachbearbeiter" zugeordnet.

```
01 <definitions id="example" ...>
02   ...
03   <extension mustUnderstand="false" definition="bpmn:staffAssignment"/>
04   ...
05   <userTask name="Reservierung bearbeiten" id="reservierungBearbeiten">
06     ...
07     <staffAssignment>R = "Sachbearbeiter" AND OE = ↩
08     "Abwicklungsabteilung"</bpmn:staffAssignment>
09   </userTask>
10   ...
11 </definitions>
```

Listing 4.5: Bearbeiterzuordnung mit Hilfe des eben definierten Attributs

Wird dem Modellierer über die Templates hinaus die Möglichkeit gegeben, die Bearbeiterzuordnung durch einen Freitext zu beschreiben, kann dieser auf dieselben Möglichkeiten in das Modell integriert werden, wie auch die Templates (`documentation`, `Text-Annotation` oder neues Attribut).

Für die Templates ist eine Speicherung in einem neuen Attribut die beste Variante. Durch die Speicherung in einem eigenen Attribut ist klar ersichtlich, dass es sich um eine Bearbeiterzuordnung handelt. Bei einer Speicherung als `documentation` oder `Text-Annotation` ist dies nicht der Fall. Zudem ist eine `Text-Annotation` ein im Diagramm sichtbares Element. Dadurch leidet die Übersichtlichkeit.

Durch die in der BPMN standardmäßig vorhandenen Optionen, einem Element einen Freitext zuzuordnen (`Text-Annotation` und `documentation`), kann man einem Fachmodellierer nicht verbieten, darauf zurückzugreifen. Eine zusätzliche Möglichkeit, die Bearbeiterzuordnung per Freitext über ein weiteres Attribut der `Activities` zu definieren, ist deshalb nicht sinnvoll.

4.2 Daten

In einem Geschäftsprozess werden für die einzelnen Prozessschritte Daten als Eingabe benötigt. Zudem werden Daten von ihnen generiert. Diese Daten sollen vom Fachmodellierer modelliert

werden. Daten werden jedoch nicht nur von Prozessschritten verwendet. Auch bei Verzweigungen des Kontrollflusses wird auf Daten zugegriffen, um je nach aktuellem Zustand die richtigen Kanten zu aktivieren. Nachdem beispielsweise im Prozessschritt A.2 "Kunde überprüfen" des Anwendungsszenarios (vgl. Abb. 1.2) der Kunde überprüft wurde, wird ein Attribut des Datenelements Kunde entsprechend des Ergebnisses der Überprüfung gesetzt. In der nachfolgenden Verzweigung wird auf das eben gesetzte Attribut zurückgegriffen und so die richtige Kontrollflusskante ausgewählt.

4.2.1 Sicht der Fachmodellierer

Bereits bei der fachlichen Modellierung ist es wichtig, die relevanten Datenobjekte im Prozessmodell anzugeben. Zum einen, da der Fachmodellierer im Gegensatz zum Systemmodellierer weiß, welche Daten geschrieben und gelesen werden, und zum anderen, weil die Daten gegebenenfalls für eine Simulation des Fachmodells benötigt werden. Dies gilt für die Eingabe- und Ausgabedaten des Prozesses ebenso wie für die Eingabe- und Ausgabedaten der einzelnen Prozessschritte. Im fachlichen Modell ist es insbesondere wichtig, alle diese Daten zu identifizieren und mit den Abhängigkeiten zu den Prozessschritten zu modellieren, da ansonsten spätere Rückfragen der IT-Abteilung nötig sind. Darüber hinaus ist es vorteilhaft bereits im Fachmodell, die Struktur der Daten genauer zu spezifizieren und die Abhängigkeiten zwischen den Daten zu modellieren.

Darstellung von Daten im Prozessmodell

Die Daten und deren Strukturbeschreibung müssen in ein Prozessmodell integriert werden. Da die Daten wichtige Elemente innerhalb des Prozessablaufs sind, sollten sie durch ein eigenes Element modelliert werden. Die Datenelemente können durch ein Symbol, inklusive einem sprechenden Namen modelliert werden (vgl. Abb. 4.4, "Auftrag"). Die modellierten Datenelemente können durch gerichtete Pfeile mit Prozessschritten verbunden werden. So wird deutlich gemacht, welche Daten als Eingabe verwendet werden und welche als Ausgabe generiert werden. Durch ein weiteres Symbol kann ein Datum als Input beziehungsweise Output eines ganzen Prozesses deklariert werden.

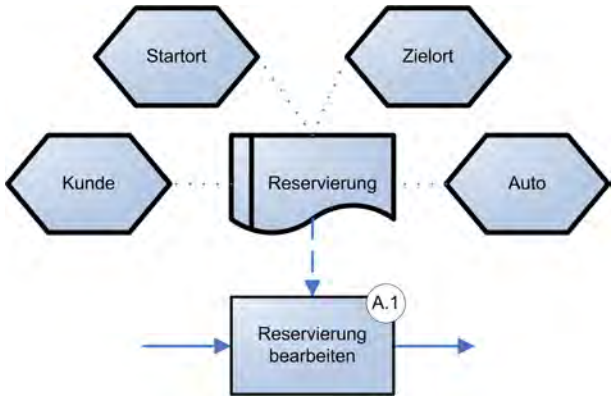


Abbildung 4.4: Datenmodellierung im Prozessmodell

Um die Daten weiter zur detaillieren und so die Struktur anzugeben, sind verschiedene Ansätze denkbar:

Grafische Beschreibung:
Eine detailliertere Beschreibung der Daten wird durch zusätzliche grafische Elemente im Prozessmodell realisiert (vgl. Abb. 4.4, "Startort"). Diese zusätzliche Beschreibung kann zu einer sehr komplexen Darstellung führen, wodurch der eigentliche Prozessablauf in den Hintergrund rückt. Werden zudem Abhängigkeiten zwischen Daten modelliert, wird das Modell noch unübersichtlicher. Die automatischen Überführungsmöglichkeiten ins Systemmodell hängen vom Formalisierungsgrad der Darstellung ab. Generell können die Elemente als Attribute angesehen und so übernommen werden. Im Systemmodell muss dann eine weitere Detaillierung erfolgen.

Freie textuelle Beschreibung:
Alternativ kann eine detaillierte Beschreibung der Daten durch einen hinterlegten Text realisiert werden. Die Datenstruktur kann über einen Freitext beschrieben werden, der erläutert um welche Art von Daten es sich handelt und aus welchen Teilen diese aufgebaut sind. Durch diese freie Beschreibung kann auch ein Modellierer ohne weitreichende Kenntnisse die Daten in Umgangssprache beschreiben. Diese freie Modellierung ist jedoch ein großer Nachteil, da der Inhalt manuell in das Systemmodell überführt werden muss. Alternativ kann auch ein Text verwendet werden, der festen Regeln folgen muss. Dies vereinfacht die Überführung ins Systemmodell, erschwert jedoch die Arbeit für den Fachmodellierer.

Standardisierte Notation:
Es existieren bereits verschiedene standardisierte Notationen um Daten und ihre Struktur zu modellieren. Einige Modellierer beschreiben etwa über UML Klassendiagramme Daten sowie deren Beziehungen zueinander grafisch. Andere hingegen beschreiben die Daten textuell, beispielsweise durch XML beziehungsweise XML-Schema. Beide Notationen sind Standards, die sich über viele Jahre bewährt haben und in der Praxis eingesetzt werden. Die Werkzeugunterstützung ist für beide Notationen gut. Die Notationen sind formal und können dadurch leicht in das Systemmodell überführt werden. Wenn die Prozessmodellierung nicht auf der grünen Wiese beginnt, ist es zudem denkbar, dass eine Modell der Daten in einem UML Klassendiagramm bereits vorliegt. Im Prozessmodell müssen die Daten dann nur noch durch eine Referenz auf die externe Definition integriert werden.

Fazit:
Daten sind wichtige Elemente innerhalb des Prozessablaufs. Deshalb sollten sie im Prozessmodell durch eine einfache Grafik visualisiert werden, beschriftet mit einem sprechenden Namen. Für eine detailliertere Beschreibung der Daten eignet sich am besten die Verwendung einer standardisierten Notation. Durch eine Referenz des Datenelements im Prozessmodell auf die externe Beschreibung wird der Zusammenhang deutlich gemacht (siehe Abb. 4.5). Weniger versierten Modellierern sollte darüber hinaus die Möglichkeit gegeben werden, ein Datenelement mit Hilfe eines Freitextes genauer zu beschreiben. Im Folgenden wird untersucht, wie sich dieser Ansatz mit der BPMN umsetzen lässt.

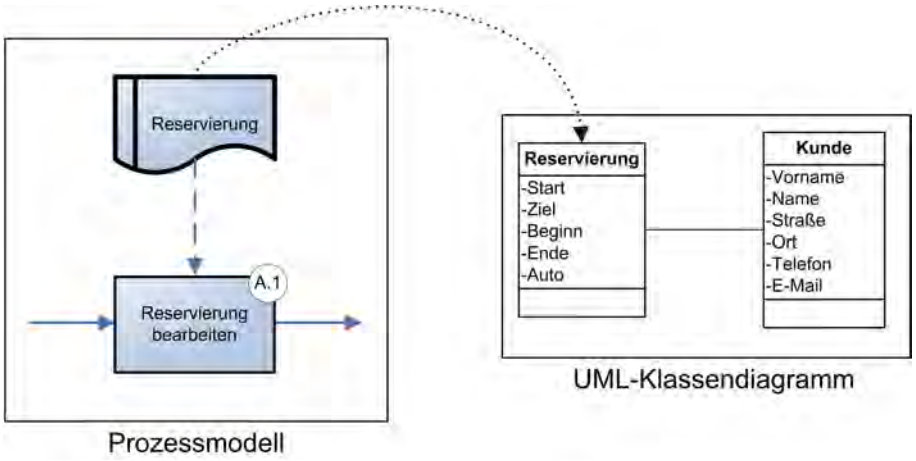


Abbildung 4.5: Referenz vom Prozessmodell in ein UML Klassendiagramm

4.2.2 Umsetzung in BPMN

In der Version 1.2 konzentrierte sich die BPMN auf den Sequenzfluss. Alle weiteren Aspekte der Prozessausführung wurden lediglich beiläufig behandelt. Auch die im Prozessablauf generierten und verwendeten Daten wurden nur am Rande betrachtet. Datenobjekte waren lediglich Artefakte, die keinen Einfluss auf den Ablauf eines Prozesses hatten. In der BPMN 2.0 erfahren die Daten eine Aufwertung. Dies liegt insbesondere darin begründet, dass für die direkte Ausführung der BPMN die Berücksichtigung der Daten essentiell ist. Den Daten wurde im Entwurf der neuen Spezifikation nun eine eigene Kategorie eingeräumt.

Im Kapitel 3.2.1 wurden die neuen Datenelemente der BPMN 2.0, ihre Visualisierung sowie ihre Verwendung bereits vorgestellt. Der erste Schritt zur Modellierung von Daten im Fachmodell ist, die Daten zu identifizieren und mit Hilfe dieser Elemente grafisch im Prozessmodell zu integrieren. Durch die Vergabe von sprechenden Namen für die einzelnen Datenobjekte wird dem Betrachter auf einen Blick deutlich gemacht, um welche Daten es sich handelt. Zusätzlich kann jedem Datenelement optional ein Status zugewiesen werden, der deutlich macht, in welchem Zustand es sich derzeit befindet. Der Status wird durch einen einfachen String angegeben, eine genauere Definition erfolgt in der Spezifikation nicht (vgl. Abb. 4.6, "[unbezahlt]"). Die Definition potentieller Werte und die hinter den Zuständen stehende Semantik wird den Werkzeugherstellern überlassen. Durch gerichtete Pfeile, die ein Datenelement mit einem Prozessschritt verbinden, wird deutlich, welcher Prozessschritt welche Datenelemente liest beziehungsweise schreibt. Abbildung 4.6 zeigt den Sub-Prozess "Abrechnung" mit den modellierten Datenelementen (vgl. Abb. 1.4).

Über die visuell sichtbare, grobe Beschreibung der Datenelemente hinaus, ist es im Zuge des Frontloadings sinnvoll, schon im fachlichen Modell die Struktur der Datenelemente genauer zu beschreiben. Die Spezifikation der BPMN bietet jedoch keine eigene Beschreibungssprache für die Struktur von Daten an. Formalisierte Referenzpunkte ermöglichen jedoch, extern definierte Datenstrukturen innerhalb des Prozessdiagramms zu referenzieren. Somit können innerhalb eines BPMN-Modells verschiedene Datenstrukturen verwendet werden. Um die Daten zu beschreiben,

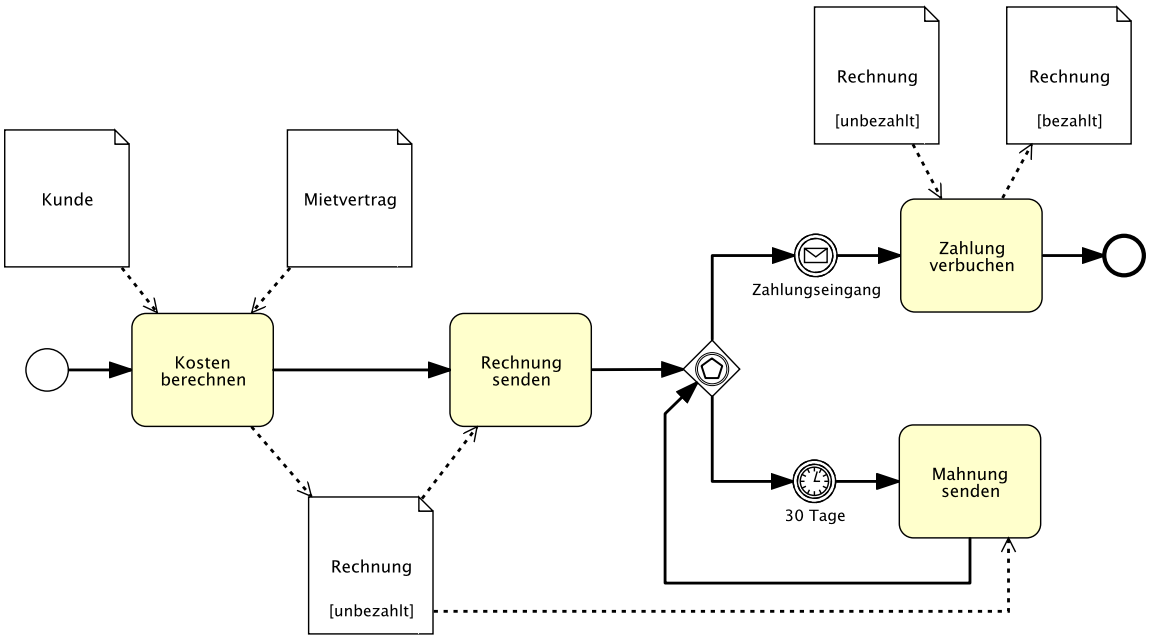


Abbildung 4.6: Abrechnungsprozess mit modellierten Datenelementen

kann eine beliebige Notation verwendet werden. Denkbar wären zum Beispiel UML Klassendiagramme, XML-Schema oder Java für diesen Zweck zu verwenden. In der BPMN-Spezifikation wird als Standardeinstellung zur externen Modellierung der Datenstrukturen XML-Schema verwendet. Will man eine andere Notation verwenden, muss dies explizit angegeben werden.

Eine formal korrekte Definition der Datenstrukturen ist keine einfache Aufgabe. Nicht jeder Modellierer des Fachmodells hat dazu genügend IT-Know-How. Deshalb muss es möglich sein, alternativ zur formalen Spezifikation von Daten einen Freitext zur Beschreibung angeben zu können. Dies wird in der BPMN mit Hilfe einer `Text-Annotation` unterstützt, die mit dem zu beschreibenden Datenelement verbunden wird (siehe Abb. 4.7). Ist eine grafische Darstellung des Freitextes nicht erwünscht, kann auf das Attribut `documentation` zurückgegriffen werden. Dieses Attribut ist für jedes Element der BPMN verfügbar und ermöglicht dessen freie textuelle Beschreibung. Da für jedes Element mehrere `documentations` hinterlegt werden können, muss explizit gemacht werden, dass es sich um die Beschreibung der Datenstruktur handelt. Beispielsweise durch ein vorangestelltes Schlüsselwort wie etwa “Datenstruktur”.

Um auf die beschriebenen Datenstrukturen zum Beispiel bei Verzweigungen zu verweisen, können in der BPMN `Expressions` verwendet werden. Dieses BPMN-Element beschreibt eine natürlichsprachliche Anfrage. Diese kann an einen `Sequence Flow` oder an diverse `Gateways` angehängt werden. Der Fachmodellierer hat so die Möglichkeit, den Bezug zu den im Prozesskontext vorhandenen Daten zu erläutern. Das `Gateway` nach dem Prozessschritt A.2 “Kunde überprüfen” (vgl. Abb. 1.2), kann bei den beiden ausgehenden `Sequence Flows` jeweils eine `Expression` mit dem Text “Attribut ’angenommen’ des Kunden gesetzt” beziehungsweise “Attribut ’angenommen’ des

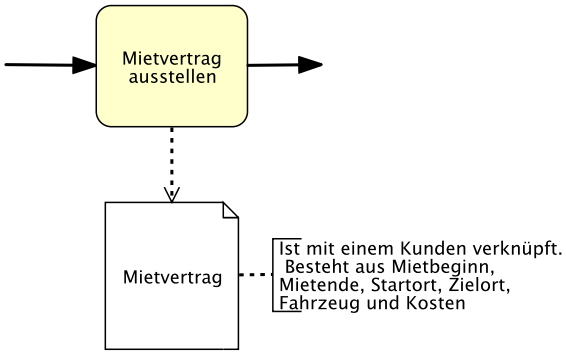


Abbildung 4.7: Datenelement mit beschreibender Text-Annotation

Kunden nicht gesetzt" hinterlegt werden.

4.3 Ausnahmebehandlung

Bei der Prozessausführung kann es zu unerwarteten Ereignissen kommen, die den Prozessablauf von seinem gewünschten Kontrollfluss abbringen. Man sprich von Ausnahmen. Diese können sowohl technischer als auch fachlicher Natur sein. Ein Beispiel für eine technische Ausnahme wäre ein fehlgeschlagener Aufruf eines Webservices. Die Stornierung einer Reservierung hingegen ist eine fachliche Ausnahme. Wie die beiden Beispiele bereits zeigen, unterscheidet sich nicht nur die Art der Ausnahmen, sondern auch ihr Gültigkeitsbereich. Eine Ausnahme kann lediglich in einem einzelnen Prozesschritt relevant sein, wie ein fehlgeschlagener Aufruf eines Webservices, oder sich über mehrere Prozessschritte hinweg erstrecken, wie die mögliche Stornierung einer Reservierung. Tritt eine Ausnahme auf, muss auf diese adäquat reagiert werden.

4.3.1 Sicht der Fachmodellierer

Die Modellierer des fachlichen Modells sind mit dem Ablauf des Prozesses bestens vertraut. Sie wissen, wo in einem Prozess Situationen eintreten können, die so nicht dem optimalen Prozessablauf, dem "Happy Path", entsprechen. Dieses Wissen muss bereits in dieser frühen Phase der Modellierung in das Prozessmodell einfließen. Die möglichen Ausnahmen lassen sich in verschiedene Kategorien einteilen [Sil09].

Ausnahmekategorien

Interne Ausnahmen im Prozess:
Der Großteil der Ausnahmen innerhalb des Prozessablaufs sind interne Ausnahmen. Dies bedeutet, dass die Ausnahme von einem Prozessschritt entdeckt wird, der normal beendet wird. Ein

Beispiel hierfür ist der Prozesschritt 1.2 "Kunde überprüfen". Im Normalfall wird der Kunde akzeptiert, unter Umständen jedoch auch abgelehnt. Der Prozessschritt wird allerdings in beiden Fällen normal beendet und eine Entscheidung über den korrekten weiteren Kontrollfluss wird später getroffen.

Ausnahme bei Benutzerinteraktion:
Dies ist lediglich eine Variante der internen Ausnahmen im Prozess. Der Prozessschritt wurde nicht beendet, bevor die Ausnahme festgestellt wird, sondern ist noch am Laufen, während durch einen Benutzer eine Ausnahme entdeckt wird. Die Folge ist, dass der Prozesschritt entweder unterbrochen wird oder typischerweise ein paralleler Zweig zur Ausnahmebehandlung geöffnet wird. Im Anwendungsszenario könnte während der Ausführung des Prozesschrittes 1.1 "Reservierung bearbeiten" der Kunde sich telefonisch melden, da er einen Fehler bei der Reservierung festgestellt hat (beispielsweise Angabe eines falschen Datums) und diesen korrigieren lassen möchte.

Fehler:
Ein Fehler ist eine Ausnahme, die bei einem automatischen Prozessschritt auftritt, wenn dieser nicht korrekt arbeiten kann. Dies kann zum Beispiel geschehen, wenn der aufgerufene Webservice aktuell nicht erreichbar ist oder die übergebenen Daten eine korrekte Ausführung nicht ermöglichen.

Deadlines:
Einige Prozessschritte müssen innerhalb einer festgelegten Zeitspanne oder bis zu einem Enddatum abgeschlossen sein. Ist dies nicht der Fall, muss diese Ausnahme abgefangen werden. Der Prozessschritt 2.4 "Fahrzeug zurückbuchen" muss beispielsweise innerhalb von maximal einer Stunde abgeschlossen sein.

Unerwartete externe Ausnahmen:
Durch unerwartet eingehende Nachrichten von Geschäftspartnern muss der Prozessablauf unterbrochen werden können oder der Start eines parallelen Pfades eingeleitet werden. Ein Kunde könnte zum Beispiel seine bereits getätigte Reservierung stornieren. Der laufende Prozess muss beendet und möglichst alle bereits erledigten Arbeiten rückgängig gemacht werden.

Ausnahmen bei angeforderten Nachrichten:
Stellt ein Prozess eine Anfrage an einen Geschäftspartner, erwartet er eine gewisse Antwort. Die Antwort kann jedoch unter Umständen anders als erwartet ausfallen. Es ist zum Beispiel möglich, dass eine Antwort komplett ausbleibt oder statt der erwarteten E-Mail mit einem Angebot, eine E-Mail mit einer Urlaubsbenachrichtigung ankommt.

Zusammenfassend sind also unterschiedliche Reaktionen beim Auftreten einer Ausnahme sinnvoll. Tritt eine Ausnahme auf, kann dies zur Folge haben, dass der aktuelle Prozessablauf gestoppt wird und ein alternativer Pfad zur Ausführung kommt. Unter Umständen müssen dann bereits abgearbeitete Schritte zurückgesetzt werden. Denkbar ist auch, dass der normale Prozessablauf nicht unterbrochen wird, sondern lediglich ein weiterer, paralleler Prozesspfad zur Ausführung kommt. Ausnahmen können einzelne Prozessschritte, mehrere Prozessschritte oder den kompletten Prozess betreffen.

Darstellung der Ausnahmen im Prozessmodell

Modell:
Um die verschiedenen Arten von Ausnahmen im Prozessmodell zu integrieren, können verschiedene Ansätze gewählt werden. Da es sich bei den Ausnahmen um Elemente innerhalb des Prozessablaufs handelt, die den Kontrollfluss direkt beeinflussen, ist es sinnvoll, diese auch grafisch, durch Elemente, die direkt in den Kontrollfluss des Prozesses eingebettet werden, zu veranschaulichen. *Interne Ausnahmen* im Prozess lassen sich leicht durch ein grafisches Element nach dem entsprechenden Prozessschritt modellieren, das den Kontrollfluss aufteilt. Um klar zu machen, in welchen Fällen welcher Weg genommen wird, ist es zweckmäßig das Verzweigungselement mit einer Frage zu beschriften. Die ausgehenden Kontrollflusskanten werden mit den potentiellen Antwortmöglichkeiten beschriftet (siehe Abb. 4.8)

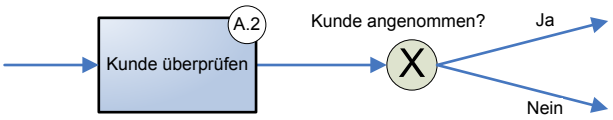


Abbildung 4.8: Grafische Modellierung interner Ausnahmen im Prozess

Um weitere Ausnahmen, wie *Deadlines* oder *Ausnahmen bei Benutzerinteraktionen*, grafisch modellieren zu können, müssen über den Standard-Kontrollfluss hinaus weitere Pfeilarten definiert werden. Deren Darstellung kann durch eine andere Linienart, andere Enden oder eingebettete Symbole vom normalen Kontrollfluss differenziert werden. Darüber hinaus muss eine unterbrechende von einer nicht unterbrechenden Variante unterschieden werden, um entweder einen parallelen oder einen alternativen Pfad darstellen zu können. Um *unerwartete externe Ausnahmen* oder *Fehler* modellieren zu können, müssen darüber hinaus weitere Elemente definiert werden, die eine Ausnahmebehandlung über mehrere Prozessschritte hinweg zulassen. Durch die vielen grafischen Elemente und aufgrund der vielen potentiellen Ausnahmefälle innerhalb eines Prozesses, ist eine grafische Modellierung sehr komplex und würde einen Fachmodellierer schnell überfordern. Die Überführung ins Systemmodell könnte jedoch, unter der Bedingung, dass korrekt modelliert wurde, ohne großen Aufwand durchgeführt werden.

Freie textuelle Beschreibung:
Eine weitere Möglichkeit, Ausnahmen zu beschreiben, ist eine freie textuelle Beschreibung, die an den jeweiligen Stellen, an der die Ausnahme auftreten kann, hinterlegt wird. Dem Prozessschritt 2.4 "Fahrzeug zurückbuchen" kann zum Beispiel der Text "Ausnahme: Prozessschritt muss innerhalb von 60 Minuten nach Beginn bearbeitet sein, ansonsten wird der Vorgesetzte informiert" zugeordnet werden. Diese Art der Beschreibung ermöglicht es dem Modellierer, relativ frei ohne viele Einschränkungen seine Erfahrungen in das Prozessmodell zu integrieren. Dies kann auch von einem unerfahrenen Modellierer leicht erledigt werden. Die Überführung in das Systemmodell muss jedoch vollständig manuell erfolgen, da die Texte keinerlei formalen Ansprüchen genügen müssen.

Strukturierte textuelle Beschreibung:
Um den Nachteil der manuellen Überführung zumindest abzuschwächen, kann eine strukturierte textuelle Beschreibung verwendet werden. Dem Modellierer wird dabei eine feste Struktur

vorgegeben, die er befüllen muss. Durch diese feste Struktur ist die Transformation ins Systemmodell einfacher realisierbar. Wird der Modellierer durch eine gute Eingabemaske unterstützt, kann die Ausnahmebehandlung durch eine strukturierte textuelle Beschreibung auch von einem unerfahrenen Modellierer erledigt werden. Eine denkbare Struktur ist eine Aufteilung in Auftrittsbedingung, Beschreibung, Konsequenz und Referenz. Die Auftrittsbedingung gibt an, wann eine Ausnahme ausgelöst wird. Die Angabe kann durch einen Freitext erfolgen. Alternativ und formaler kann aber auch eine formale Grammatik, ähnlich zu der in Kapitel 4.1.1 bei der Bearbeiterzuordnung beschriebenen, hinterlegt werden. Im eben aufgezeigten Beispiel (siehe Abb. 4.9) könnte die Auftrittsbedingung "Jetzt $>$ Start $+$ 60 Minuten" lauten. Mit Hilfe der Beschreibung kann der Modellierer die Ausnahme genauer spezifizieren (im Beispiel "Ausnahme tritt auf, wenn der Prozessschritt nicht innerhalb von 60 Minuten nach Beginn fertig bearbeitet wurde"). Die Konsequenz legt fest, was beim Auftreten der Ausnahme passieren soll. Im hier besprochenen Fall wäre dies "Vorgesetzten des Bearbeiters informieren, der Prozessschritt läuft normal weiter". Die Referenz legt fest, welche Elemente innerhalb des Prozesses betroffen sind. Im Beispiel wäre dies der Prozessschritt 2.4 "Fahrzeug zurückbuchen". Unter Umständen kann dies aber auch mehrere Prozessschritte betreffen, wie zum Beispiel bei einer eingehenden Stornierung einer Reservierung.

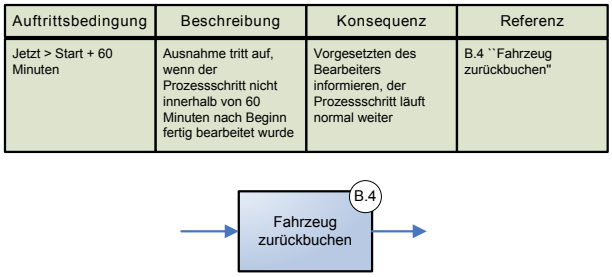


Abbildung 4.9: Beschreibung einer Ausnahme durch strukturierte textuelle Dokumentation

Fazit:
Eine komplette grafische Modellierung aller Ausnahmen würde einen Modellierer aus der Fachabteilung sicherlich überfordern. Zudem wäre das entstandene Prozessmodell so komplex, dass es nicht mehr intuitiv und schnell erfassbar wäre. Da eine strukturierte textuelle Beschreibung für die Modellierer einfacher zu handhaben ist und das Prozessmodell dadurch nicht überfrachtet wird, eignet sich diese besser. Aufgrund der besseren Überführbarkeit ins Systemmodell ist sie einer freien textuellen Beschreibung vorzuziehen. Eine Besonderheit bilden die *internen Ausnahmen im Prozess*. Da dies Ausnahmen sind, die im Alltag des Fachbereichs regelmäßig vorkommen und den Hauptablauf des Prozesses so häufig direkt beeinflussen, ist es vorteilhaft diese, wie oben dargestellt, bereits im fachlichen Modell grafisch zu modellieren.

4.3.2 Umsetzung in BPMN

Wie bereits in Kapitel 3.2.2 vorgestellt, existiert in der BPMN eine Vielzahl an `Events`, um im Prozessablauf auftretende Ausnahmen und die entsprechende Reaktion darauf explizit zu modellieren. Um den Modellierer aus dem Fachbereich jedoch nicht zu überfordern, muss eine vernünftige Auswahl für das fachliche Modell getroffen werden.

Eine grafische Modellierung der *internen Ausnahmen* ist jedoch, wie bereits besprochen, sinnvoll. Nachdem eine `Activity` bearbeitet wurde, kann der ausgehende Kontrollfluss durch ein `Gateway` verzweigt werden. Die Verzweigungsbedingung bezieht sich dabei auf eine in der vorhergehenden `Activity` getroffene Entscheidung. Dies kann durch die Benennung mit einer Frage deutlich gemacht werden. Die ausgehenden Kontrollflusskanten repräsentieren die verschiedenen Antwortmöglichkeiten (siehe Abb. 4.10).

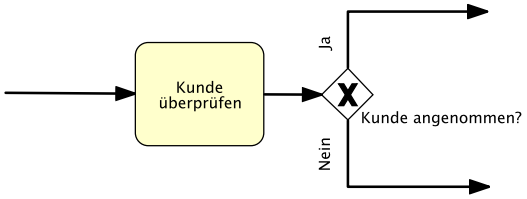


Abbildung 4.10: Modellierung einer internen Ausnahmen im BPMN-Diagramm

Die weiteren Ausnahmen können durch eine strukturierte textuelle Beschreibung erfasst werden. Die Beschreibung kann wiederum in einer `Text-Annotation` oder im Attribut `documentation` des jeweiligen Elements hinterlegt werden. Diese beiden Alternativen bieten jedoch jeweils nur die Option, einen freien Text zu hinterlegen, eine Möglichkeit eine Struktur vorzugeben ist nicht vorhanden. Durch die Definition eigener Attribute (siehe Kapitel 4.1.2) kann jedoch eine Struktur, wie in Kapitel 4.3.1 beschrieben, umgesetzt werden. Wird die Eingabe der Ausnahmen zudem von einem grafischen Assistenten unterstützt, ähnlich dem in Abb. 4.2, erleichtert dies die Arbeit für den Fachmodellierer.

Gilt eine Ausnahme über mehrere Prozessschritte hinweg, können diese durch eine Gruppierung (siehe Abb. 3.5a) zusammengefasst werden, um deutlicher zu machen, dass die Ausnahme für alle in der Gruppierung enthaltenen Schritte Gültigkeit hat.

4.4 Flexibilitätsstellen

Die Abläufe und Anforderungen eines Unternehmens ändern sich im Laufe der Zeit. Gründe dafür können in veränderten Abläufen innerhalb des Unternehmens oder im veränderten wirtschaftlichen Umfeld liegen. Um auf solche Änderungen schnell und flexibel reagieren zu können, müssen sich die modellierten Geschäftsprozesse leicht an die geänderten Rahmenbedingungen anpassen lassen. Werden die Stellen, an denen Flexibilität erwartet wird, schon im Fachmodell markiert, erhöht dies die Flexibilität bei der Implementierung, woduch auftretende Änderungen leichter interpretiert werden können.

4.4.1 Sicht der Fachmodellierer

Im Laufe der Zeit ändert sich ein Geschäftsprozess. Veränderungen des Geschäftsprozesses können sich aus unterschiedlichen Gründen ergeben [AJ00]. Die Erfahrung von Fachmodellierern

kann dazu genutzt werden, früh während der Geschäftsprozessmodellierung potentielle Stellen zu markieren, die später flexibel implementiert werden sollen. Wird ein Unternehmen etwa häufig umstrukturiert, kann er dies im Fachmodell vermerken. Bei der späteren Implementierung der Bearbeiterzuordnung wird dann auf leichte Anpassbarkeit geachtet. Der Modellierer des Systemmodells bekommt also die Möglichkeit diese Stellen explizit zu behandeln, um auf Änderungen flexibler reagieren zu können.

Ursachen für Flexibilitätsstellen

Änderungen in einem Geschäftsprozess, die eine Veränderung des Modells nach sich ziehen, können durch unterschiedliche Gründe ausgelöst werden:

Umfeld des Geschäftsprozesses:
Geschäftsprozesse sind nicht starr, sondern können im Laufe der Zeit Veränderungen erfahren, die aus einem geänderten Umfeld resultieren. Durch das Aufkommen von Navigationsgeräten kann dem Kunden beim Mieten eines Autos zum Beispiel optional die Miete eines solchen Gerätes angeboten werden. Beim Auswählen eines Fahrzeuges muss dann berücksichtigt werden, dass in dem gewählten Fahrzeug ein Navigationsgerät vorhanden ist.

Rechtsgrundlage:
Durch den Gesetzgeber werden regelmäßig neue Gesetze erlassen, Gesetze geändert oder Gesetze gestrichen. Veränderungen der Rechtsgrundlage müssen zeitnah in den Geschäftsprozess integriert werden. Wird zum Beispiel ein Gesetz verabschiedet, dass das Fahren eines Mietautos erst mit dem Erreichen des 25. Lebensjahres gestattet, muss dies im Prozessschritt [A] "Reservierung" beachtet werden. Ist ein Kunde jünger als 25 Jahre, muss er abgelehnt werden.

Technologie:
Das Aufkommen neuer Technologien oder eine Änderung in der bestehenden technische Infrastruktur haben Auswirkungen auf die Geschäftsprozesse. Der durch die Schufa bereitgestellte Web-Service zur Beauskunftung von Kunden kann zum Beispiel auf einen anderen Server migriert werden. Dies hat bei einer starren Implementierung zur Folge, dass der Aufruf des Web-Services innerhalb des Geschäftsprozesses geändert werden muss.

Typen von Flexibilitätsstellen

Neben der Art von konkreten Änderungen wird nun beschrieben, an welchen Stellen im Prozess Änderungen auftreten können:

Datenflexibilität:
Durch die Entwicklungen in der Automobilbranche und neue Gesetze wird zum Beispiel das Datenobjekt "Fahrzeug" oft geändert. Das heißt hier ist eine Flexibilitätsmarkierung sinnvoll, um solche Änderungen später einfach realisieren zu können. Die Einführung der Umweltplaketten erfordert beispielsweise ein neues Attribut, wodurch das Datenobjekt "Fahrzeug" angepasst werden muss.

Bearbeiterflexibilität:
Im Laufe der Zeit können in einem Unternehmen neue Rollen, Kompetenzen, Gruppen oder

Organisationseinheiten (siehe Kapitel 4.1.1) hinzugefügt werden. Bestehende können abgeändert oder gelöscht werden. Denkbar ist auch eine Änderung der kompletten Organisationsstruktur des Unternehmens. All diese Änderungen beeinflussen die Zuordnung der Bearbeiter zu den einzelnen Prozessschritten [RCWR04]. Häufig vorkommende Änderungen an der Organisationsstruktur sollten deshalb möglichst flexibel implementiert werden. Dazu müssen solche Stellen bereits während der Modellierung durch den Fachmodellierer markiert werden, um später bei der Implementierung entsprechend darauf zu reagieren.

Flexibilität an Verzweigungsbedingung:
Durch veränderte Rechtsgrundlagen oder Veränderungen im Umfeld des Geschäftsprozesses werden häufig die Verzweigungen des Kontrollflusses beeinflusst. Wird vom Gesetzgeber etwa ein Gesetz verabschiedet, welches das Anmahnen einer Rechnung erst nach einer Wartezeit von mehr als 45 Tagen erlaubt, muss die Bedingung in Prozessschritt B.5 "Abrechnung" geändert werden. Verzweigungsbedingungen, die flexibel sind, sollten deshalb vom Fachmodellierer markiert werden, um diese bei der späteren Implementierung entsprechend behandeln zu können.

Prozessschrittflexibilität:
An Prozessschritten können während der Lebensdauer eines Prozesses Änderungen auftreten, die das Löschen, Aufteilen oder Ändern dieser erforderlich machen. Im Anwendungsszenario aus Abschnitt 1.2 kann etwa der Prozessschritt A.4 "Fahrzeug auswählen" durch Änderungen des Datenobjektes "Fahrzeug" ebenfalls geändert werden müssen. Der Prozessschritt sollte deshalb vom Fachmodellierer markiert werden.

Flexibilität bei der Servicezuweisung:
Ändert sich die technische Infrastruktur, kann dies Veränderungen im Prozessablauf zur Folge haben. Wird zum Beispiel der von der Schufa angebotene Web-Service zum Abfragen der Liquidität eines Kunden auf einen anderen Server migriert, muss der Aufruf des Web-Services im Prozessmodell geändert werden, sodass der Web-Service auf dem neuen Server aufgerufen wird.

Ein Fachmodellierer markiert Stellen im Geschäftsprozess, die in der Vergangenheit oft abgeändert wurden. Diese Markierungen können dann bei der Implementierung wieder aufgegriffen werden, um entsprechende Flexibilität in der Software-Realisierung zu erreichen. Um dies durchzuführen sind verschiedene Ansätze denkbar.

Darstellung von Flexibilitätsstellen im Prozessmodell

Grafische Modellierung:
Flexibilitätsstellen im Geschäftsprozess können durch ein Symbol im Prozessmodell visualisiert werden. Für die unterschiedlichen Typen der Flexibilität werden Abwandlungen des Symbols verwendet, um diese voneinander unterscheiden zu können. Abbildung 4.11 zeigt ein Beispiel einer grafischen Modellierung von Flexibilitätsstellen. Beim Prozessschritt A.1 ist zu erwarten, dass sich die Bearbeiterzuordnung ändern wird. Der Prozessschritt wird folglich mit einem Symbol vom Typ Bearbeiterflexibilität markiert. Bei der Verzweigung nach dem Prozessschritt B.5.2 wurde die Flexibilität bei der Verzweigungsbedingung markiert.

Für den Modellierer des Fachmodells ist diese Art der Darstellung leicht verständlich. Nachdem er an einer Stelle eine Flexibilität identifiziert hat, modelliert er dort das entsprechende Symbol.

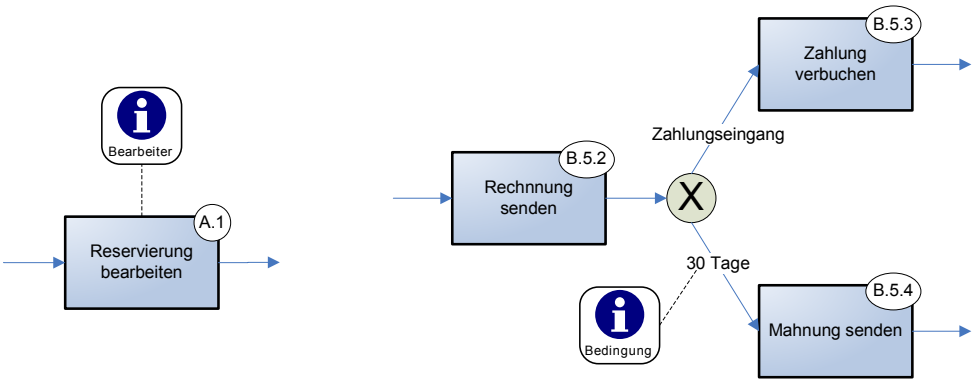


Abbildung 4.11: Grafische Modellierung von Flexibilitätsstellen

Die Symbole können in das Systemmodell übernommen werden, sodass die markierten Stellen durch die IT-Abteilung bei der Implementierung entsprechend behandelt werden können.

Freie textuelle Beschreibung:
Der Modellierer beschreibt die Flexibilitätsstellen textuell. Diese Beschreibung wird an der entsprechenden Stelle im Prozessmodell hinterlegt. Der Detaillierungsgrad bleibt dabei dem Modellierer überlassen. Er kann zum Beispiel lediglich modellieren, dass es sich um eine Bearbeiterflexibilität handelt. Er kann aber auch genau detaillieren, dass durch eine bereits in Planung befindliche Umstrukturierung des Unternehmens in Zukunft nicht mehr ein Mitarbeiter der Abwicklungsabteilung, sondern ein Kollege aus der Schadensregulierung einen bestimmten Prozessschritt bearbeiten muss.
Für den Modellierer ist diese Art der Darstellung einfach zu realisieren, da er sich an keine Vorgaben halten muss. Die Überführung in das Systemmodell muss aufgrund der nicht vorhandenen Formalisierung manuell erfolgen.

Strukturierte textuelle Beschreibung:
Um einen höheren Formalisierungsgrad zu erreichen kann eine strukturierte textuelle Beschreibung verwendet werden (vgl. Abschnitt 4.3.1).

Fazit:
Zusammenfassend ist die Modellierung durch grafische Elemente mit einer zusätzlichen freien textuellen Beschreibung zu empfehlen. Die im Fachmodell modellierten Flexibilitätsstellen können automatisch in das Systemmodell übernommen werden. Der Implementierer kann mit Hilfe dieser Informationen entsprechende Maßnahmen ergreifen. Wichtig dafür ist insbesondere die Information darüber, um welchen Typ von Flexibilität es sich handelt, da die unterschiedlichen Typen verschieden behandelt werden müssen. Innerhalb der Typen gibt es jedoch nur wenige Unterscheidungen.

Eine grafische Modellierung ist für die Modellierer einfach zu handhaben. Darüber hinaus ist durch eine Unterscheidung der Symbole leicht ersichtlich, um welchen Typ von Flexibilität es sich handelt. Um Modellierern darüber hinaus noch die Möglichkeit zu geben, detailliertere Informationen zu hinterlegen, kann ein zusätzlicher Freitext verwendet werden. Dieser kann mit den grafischen Symbolen verknüpft werden. Um die grafische Darstellung des Geschäftsprozesses

jedoch nicht unnötig zu überfrachten, sollte die Beschreibung lediglich als textuelles Attribut des Symbols gespeichert und nicht grafisch dargestellt werden.

4.4.2 Umsetzung in BPMN

In der BPMN gibt es keine speziellen Elemente, um Flexibilitätsstellen explizit zu visualisieren. Um Flexibilitätsstellen grafisch modellieren zu können, muss deshalb ein neues Element definiert werden.

In Abschnitt 3.2.1 wurden bereits die `Artifacts` vorgestellt. Diese dienen dazu, den Geschäftsprozess um weitere Informationen zu erweitern, die über den eigentlichen Kontroll- und Datenfluss hinausgehen. Um Flexibilitätsstellen zu modellieren, wird das neue `Artifact Flexibility` definiert (siehe Anhang B). Letzteres besteht aus der Angabe eines Typs und einer Beschreibung als Freitext. Als Typ können die in Kapitel 4.4.1 definierten Flexibilitätsstellen angegeben werden. Im Attribut "Text" kann darüber hinaus eine detailliertere Beschreibung in Form eines Freitextes hinterlegt werden.

Abbildung 4.12 zeigt die im BPMN realisierte Modellierung einer Flexibilitätsstelle. Die Bearbeiterzuordnung der `Activity` "Reservierung bearbeiten" ist flexibel. Die Markierung ist folglich vom Typ Bearbeiterflexibilität. Weitere Details werden im hinterlegten XML als Freitext beschrieben.

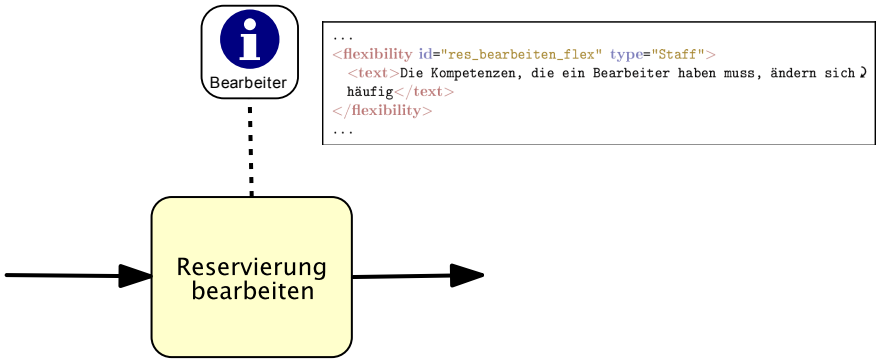


Abbildung 4.12: Flexibilitätsstelle in BPMN

4.5 Zusammenfassung

Im fachlichen Modell wurden nun aus Sicht der Fachabteilung alle relevanten Aspekte modelliert. Der Modellierer hat dabei im Zuge des Frontloadings seinen Fähigkeiten entsprechend bereits in dieser frühen Phase möglichst viele Informationen in das Modell einfließen lassen. So können spätere Rückfragen der IT-Abteilung minimiert werden. In diesem Kapitel wurden Konzepte zur Modellierung von Daten, Bearbeiterzuordnung, Ausnahmebehandlung und Flexibilitätsstellen vorgestellt. Darüber hinaus müssen auch alle anderen Anforderungen, die im Rahmen dieser

Arbeit nicht betrachtet wurden und für das fachliche Modell relevant sind, in das Prozessmodell integriert werden (vgl. Abschnitt 1.3).

Um den nun fachlich modellierten Geschäftsprozess ausführen zu können, muss dieser vervollständigt und formalisiert werden. Wie dies für die im Rahmen dieser Arbeit untersuchten Aspekte aussehen kann, wird im nächsten Kapitel vorgestellt.

5 Modellierung auf Systemebene

Das fachliche Modell wurde von Modellierern der Fachabteilung erstellt. Dabei wurden im Zuge des Frontloadings bereits möglichst viele Informationen modelliert. Das Systemmodell baut auf dem fachlichen Modell auf und erweitert dieses. Ziel ist es ein vollständiges und formales Modell des Geschäftsprozesses zu erhalten, das automatisch in ein ausführbares Modell überführt werden kann. Alle Details müssen deshalb präzise ausmodelliert werden. Es darf keinen Interpretationsspielraum mehr geben, da ein Computer damit nicht umgehen kann.

Im ersten Schritt kann eine Kopie des fachlichen Modells erstellt werden. Dabei kann die Beziehung zwischen Elementen im fachlichen Modell und im Systemmodell gespeichert werden [BBR10a, BBR10b]. So bleibt für die Fachabteilung nachvollziehbar, welche Teile des fachlichen Modells bei Änderungen im Systemmodell betroffen sind. Im nächsten Schritt kann das kopierte Modell vervollständigt und formalisiert werden.

Im Folgenden werden einzelne Aspekte, welche der Modellierer des Systemmodells betrachten muss detailliert. In Abschnitt 5.1 wird die Bearbeiterzuordnung konkretisiert. Anschließend wird in Abschnitt 5.2 auf die Verfeinerung von Datenobjekten eingegangen. Abschnitt 5.3 beschäftigt sich mit der Modellierung von Ausnahmen, Abschnitt 5.4 mit Flexibilitätsstellen. Den Abschluss bildet Abschnitt 5.5 mit einer Zusammenfassung.

5.1 Bearbeiterzuordnung

5.1.1 Sicht der Systemmodellierer

Im fachlichen Modell wurde die Bearbeiterzuordnung durch Templates vorgenommen. Bei Bedarf konnten diese um eine freie textuelle Beschreibung erweitert werden. Im Systemmodell müssen diese nun formalisiert werden. In der Regel besitzen Modellierer des Systemmodells höhere IT-Kenntnisse als ein Modellierer in der Fachabteilung. Deshalb ist es sinnvoller, diesem

die Zuordnung nicht mehr als Templates, sondern als Text mit formaler Grammatik darzustellen. Da die Templates nur eine andere Darstellungsform für die Regeln ist, bedarf dies keinerlei Überführungsarbeit.

Die Beschreibung durch Templates, die in Form der definierten Regeln abgespeichert wurden, ist bereits formal korrekt. Sie folgen einer festgelegten Syntax und können mit einer Semantik hinterlegt werden. Wurde die Beschreibung so vorgenommen, kann diese unverändert übernommen werden. Wichtig ist es aber, die Zuordnung dahingehend zu prüfen ob sie sinnvoll ist. Es ist denkbar, dass bei der Auswertung der Zuordnung keine Bearbeiter ausgewählt werden. In manchen Fällen kann dies erst zur Laufzeit festgestellt werden (z.B. bei einer Zuordnung über Daten), in manchen Fällen kann dies aber auch schon vorher festgestellt werden. Die Zuordnung K = X `AND` (G = Y `AND NOT` K = X) liefert beispielsweise immer eine leere Menge zurück. Diese Fälle müssen erkannt und beseitigt werden.

Die freie textuelle Beschreibung entspricht nicht den Anforderungen des Systemmodells. Die dort beschriebenen Sachverhalte müssen manuell überführt werden und in die vorhandene, formale Beschreibung eingearbeitet werden. Dies kann unter Umständen zu Widersprüchen führen, die beseitigt werden müssen. Werden Sachverhalte beschrieben, die nicht durch bestehende Regeln abgedeckt werden, müssen neue Regeln definiert werden.

Um auf die im Prozessablauf definierten Daten zurückgreifen zu können, ist eine komplexe Anfragesprache erforderlich. Im Fachmodell wurden die Daten unter Umständen noch nicht detailliert beschrieben, sodass zwar das Datenelement, auf das zugegriffen werden muss, referenziert wurde, nicht aber die genaue Stelle innerhalb des Elements. Da die Daten im Systemmodell nun detailliert beschrieben wurden, kann auf diese mit Hilfe einer Anfragesprache zugegriffen werden. Die Anfragesprache ist abhängig vom Format, in dem die Daten beschrieben wurden. Sind die Daten in XML definiert, sieht die Bearbeiterzuordnung eines Prozessschrittes, in dem ein Urlaubsantrag von der Personalabteilung genehmigt werden muss, der Genehmigende selbst jedoch nicht der Antragsteller sein darf, wie folgt aus:

$$\text{OE} = \text{Personalabteilung } \texttt{AND NOT } (\text{MA} = \$\text{getDataInput('antrag')/Antragsteller}\$)$$

5.1.2 Umsetzung in BPMN

Die Bearbeiterzuordnung wurde im Fachmodell durch Templates festgelegt und in einer `Text-Annotation`, einer `documentation` oder vorzugsweise in einem eigenen Attribut der `Activity` hinterlegt. Um der Formalisierung Rechnung zu tragen, ist es sinnvoll, im Systemmodell die Zuordnung in einem eigenen Attribut zu speichern. So wird deutlich, dass es sich um ein eigenes, wichtiges Element handelt. Zudem fällt die Schwierigkeit weg, eine `Text-Annotattion` oder eine `documentation` als Bearbeiterzuordnung identifizieren zu müssen, beispielsweise durch ein Schlüsselwort.

Zusätzlich wurde unter Umständen ein Freitext in einer `Text-Annotation` oder einer `documentation` hinterlegt. Ist dies der Fall, muss er in die bereits vorhandene, formale Bearbeiterzuordnung eingearbeitet werden. Die so entstandene Zuordnung kann dann im bereits vorhandenen Attribut der `Activity` gespeichert werden. Die Informationen des Freitextes sind somit nicht mehr nötig, können jedoch trotzdem zu Dokumentationszwecken im Modell behalten werden. Bei der

Überführung des Freitextes in die Regeln ist es wichtig diese abzugleichen und auf potentielle Widersprüche zu achten.

Um auf die im Prozesskontext vorhandenen Daten zugreifen zu können, kann mit einer Anfragesprache gearbeitet werden. Standardmäßig wird zur Datenbeschreibung XML eingesetzt und dementsprechend XPath als Anfragesprache verwendet (siehe Kapitel 4.2.2). Demzufolge kann auch in der Bearbeiterzuordnung mit Hilfe einer Anfragesprache auf die Daten zugegriffen werden. Die so gewonnenen Daten können auf die Zuordnung Einfluss nehmen.

5.2 Daten

5.2.1 Sicht der Systemmodellierer

Alle im Prozessablauf relevanten Daten wurden im Zuge des Frontloadings bereits im fachlichen Modell identifiziert und grafisch modelliert. Darüber hinaus wurde, soweit dies dem Modellierer möglich war, die Struktur der Datenobjekte beschrieben. Die Beschreibung liegt in einer externen Notation, etwa als XML-Schema oder als UML Klassendiagramm vor. Diese ist mit den Datenobjekten im Prozess verknüpft. Alternativ wurde die Strukturbeschreibung durch einen Freitext an das Datenobjekten angehängt. Im Systemmodell müssen nun alle vorliegenden Informationen formalisiert werden. Dazu gehört, dass die Strukur der Daten detailliert in einer formalen Notation beschrieben werden muss. Zudem müssen die Eingabe- und Ausgabedaten der einzelnen Prozessschritte, sowie die Datenbezüge der Verzweigungen formal modelliert werden.

Datenstrukturen

Wie bereits im Kapitel zur fachlichen Modellierung (siehe Kap. 4.2.1) erwähnt, soll die Beschreibung der Datenstrukturen in einer externen Notation vorgenommen werden. Da sich Standards wie UML oder XML in der Vergangenheit bewährt haben, ist es vorteilhaft, auf diese zurückzugreifen. Eine textuelle Beschreibung macht auf Systemebene keinen Sinn, da diese nicht formal ist. Vorliegender Freitext muss deshalb transformiert werden.

Die Beschreibungen der Datenstrukturen können, wenn sie bereits im fachlichen Modell in der externen Notation vorgenommen wurden, übernommen werden. Wurde ein Freitext hinterlegt, muss dieser durch einen Modellierer in ein externes Datenmodell überführt werden. Liegt sowohl ein Freitext als auch ein externes Datenmodell vor, müssen die Informationen des Freitextes in das Datenmodell integriert werden. Dabei ist zu beachten, dass das so entstandene Datenmodell widerspruchsfrei ist und keine Redundanzen enthält.

Das externe Datenmodell muss im nächsten Schritt weiter verfeinert werden. Dazu werden die Abhängigkeiten zwischen den Daten, wie Beziehungen zwischen den Daten oder Vererbungen, modelliert. Zudem sollten die Daten durch die Angabe der einzelnen Attribute und ihrer Datentypen weiter verfeinert werden. Dazu ist unter Umständen eine Rücksprache mit der Fachabteilung notwendig, um an die nötigen Informationen zu gelangen. Von der Fachabteilung modellierte Attribute können zudem verfeinert werden. Hatte das Datenobjekt “Kunde” im fachlichen Modell

noch das Attribut “Adresse”, kann dies nun weiter verfeinert werden in die Attribute “Straße”, “Hausnummer”, “Postleitzahl” und “Ort”, denen jeweils ein passender Datentyp zugewiesen wird.

Bestenfalls liegt bereits ein Datenmodell vor, das in einem anderen Kontext erstellt wurde. Ist dies der Fall und wurde das vorliegende Datenmodell nicht bereits von der Fachabteilung identifiziert und eingebunden, so muss nun geprüft werden, ob das Modell vollständig im Bezug zu den Anforderungen der Fachabteilung ist. Gegebenenfalls muss das Modell entsprechend angepasst werden. Ist dies gewährleistet, können die bereits modellierten Datenobjekte innerhalb des Prozesses mit den entsprechenden Objekten im externen Modell verknüpft werden.

Datenzugriff

Um im Prozessablauf auf die nun vollständig und formal definierten Daten zugreifen zu können, wird eine Abfragesprache benötigt. Üblicherweise liegt für jede Strukturbeschreibungssprache eine passende Abfragesprache vor. Um Informationen aus einem UML Diagramm abzufragen, kann beispielsweise die Object Constraint Language (OCL) [OMG06] verwendet werden, für XML-Dokumente existiert mit der XML Path Language (XPath) [W3C07] eine Anfragesprache.

Nun können die Ein- und Ausgabedaten der einzelnen Prozessschritte genauer definiert werden. Mit Hilfe der passenden Abfragesprache kann auf die einzelnen Daten zugegriffen werden. Dadurch kann präzise beschrieben werden, welcher Teil eines Datenobjektes als Eingabe dient und wie genau die Ausgabe erfolgt. Um zum Beispiel die Kosten eines Mietvertrages zu berechnen, ist vom Datenobjekt Kunde lediglich die Information relevant, ob es sich um einen Goldkunden handelt oder nicht. Diese kann mit Hilfe einer Anfragesprache vom Kundenobjekt selektiert werden. Darüber hinaus muss spezifiziert werden können, was passiert, wenn Eingabedaten eines Prozessschrittes nicht vorliegen. Denkbar wäre, dass die Dateneingaben lediglich optional sind und die Ausführung des Prozessschrittes trotz des Fehlens angestoßen wird. Es besteht aber auch die Möglichkeit, dass der Prozessschritt ohne die Daten zu keinem Ergebnis gelangen kann, weshalb der Schritt erst gestartet werden kann, wenn die fehlenden Daten vorliegen. Des weiteren ist es unter Umständen sinnvoll, bei einem Prozessschritt alternative Eingabedaten definieren zu können. Deren Vorhandensein wird nacheinander ausgewertet. Der Prozessschritt wird gestartet, sobald ein Eingabedatum vorliegt, welches dann als Eingabe verwendet wird. Die alternativen Eingabedaten werden nicht weiter berücksichtigt.

Daten dienen jedoch nicht nur als Aus- und Eingabe von Prozessschritten. Auch bei Verzweigungen des Kontrollflusses wird auf Datenobjekte zugegriffen, um zu entscheiden, welcher Weg genommen werden soll. Die Beschriftung eines Verzweigungsknotens im fachlichen Modell zeigt diesen Zusammenhang bereits auf. Im Systemmodell muss dieser weiter konkretisiert werden. Der Prozessschritt “Kunde überprüfen” verändert beispielsweise das Datenobjekt “Kunde” und setzt eine boolesche Variable auf “true” (entspricht angenommen) oder “false” (entspricht abgelehnt). Auf die detaillierte Struktur der Daten kann wiederum über die passende Anfragesprache zugegriffen werden. Der Anfrageausdruck kann im Verzweigungsknoten oder den ausgehenden Kontrollflusskanten gespeichert werden.

5.2.2 Umsetzung in BPMN

Im fachlichen Modell wurden bereits alle relevanten Datenobjekte mit ihren Beziehungen zu den einzelnen `Activities` im BPMN-Diagramm modelliert. Darüber hinaus wurde die Struktur bereits möglichst detailliert in einem Freitext oder wenn möglich in einem externen Modell beschrieben. Die Struktur muss nun weiter detailliert werden und eine Möglichkeit bereitgestellt werden, Abfragen auf die Daten auszuführen.

Wie bereits in Kapitel 4.2.2 erläutert, bietet die BPMN Referenzpunkte an, um auf eine externe Beschreibung der Datenstruktur zu verweisen. Der erste Schritt für den Modellierer des Systemmodells ist es, die Struktur der Datenobjekte in einem solchen externen Modell vollständig, das heißt, mit allen Attributen, den jeweiligen Datentypen und den Beziehungen zwischen den Datenobjekten, zu beschreiben. Informationen aus dem Fachmodell, die lediglich in einem Freitext vorliegen, müssen in diese Darstellung überführt werden. Reichen die vorliegenden Informationen für eine vollständige Beschreibung nicht aus, muss der Modellierer Rücksprache mit der Fachabteilung halten. Die Datenstruktur kann dabei in einer beliebigen Sprache beschrieben werden. Die verwendete Sprache kann in einem Attribut des Prozesses angegeben werden, standardmäßig wird XML-Schema verwendet. Die für den Prozess festgelegte Strukturbeschreibungssprache kann jedoch von jedem einzelnen Datenobjekt überschrieben werden, sodass es theoretisch möglich ist, die Struktur jedes Datenobjektes in einer anderen Sprache zu beschreiben.

Um Daten und deren Ein- und Ausgabe aus `Activities` zu modellieren, bietet die BPMN folgende Konstrukte an: `Data Objects`, `ItemDefinition`, `Properties`, `Data Inputs`, `Data Outputs`, `Messages`, `Input Sets`, `Output Sets` und `Data Associations`. Diese werden im folgenden genauer erläutert.

Einige Elemente in der BPMN speichern oder verarbeiten Objekte während der Ausführung des Prozesses. Diese Elemente werden `Item-Aware Elements` genannt. Sie sind vergleichbar mit Variablen in einer üblichen imperativen Programmiersprache wie beispielsweise Java. Als `Item-Aware Elements` sind in der BPMN `Data Objects`, `Properties`, `Data Inputs`, `Data Outputs` und `Messages` definiert. Jedes dieser Elemente hat eine `ItemDefinition`, in der die Datenstruktur beschrieben wird und einen `DataState`, mit dem beschrieben werden kann, in welchem Zustand sich das Objekt befindet. Mit Hilfe der `ItemDefinition` wird auf die extern definierte Struktur des Datenelements verwiesen. Listing 5.1 zeigt, wie die Definition eines `Data Objects` (Zeile 10) mit einer Verknüpfung zu einem externen Diagramm über eine `ItemDefinition` (Zeile 8) in XML realisiert wird. Um auf die extern definierte Struktur verweisen zu können, muss diese zuerst importiert werden (Zeile 5-7).

Das grundlegende Konstrukt für die Datenmodellierung in einem Prozess ist das `Data Object` Element. Dieses hat einen wohldefinierten Lebenszyklus, der die Sichtbarkeit regelt. `Data Objects` sind in einem Prozessdiagramm visualisiert und müssen immer in einem `Process` oder `Sub-Process` enthalten sein. Sie haben ein Attribut `isCollection`, das anzeigt, ob das `Data Object` ein einzelnes Element, oder eine Sammlung von Elementen ist. Dies wird auch grafisch sichtbar gemacht (siehe Abb. 3.2b. `Data Objects` können zudem optional ein `DateState` Element referenzieren, das angibt, in welchem Status sich die Daten befinden. Der Status besteht dabei lediglich aus einem String. Es bleibt dem Modellierer überlassen, für seinen Anwendungszweck sinnvolle Status zu definieren. Innerhalb eines Prozessdiagramms können `Data Objects` mehrmals auftreten. Dabei wird bei jedem Auftreten dieselbe Instanz referenziert. Diese Option wurde

```
01 <definitions ...
02              typeLanguage="http://www.w3.org/2001/XMLSchema"
03              xmlns:sd="http://www.daimler.com/sample">
04 ...
05 <import importType="http://www.w3.org/2001/XMLSchema"
06         location="sample.xsd"
07         namespace="http://www.daimler.com/sample">
08 <itemDefinition id="kundeStruktur" structure="sd:kunde"/>
09 ...
10 <dataObject id="kunde" name="Kunde" itemSubjectRef="kundeStruktur"/>
```

Listing 5.1: Referenzierung eines externen Datenmodells

geschaffen, um die Prozesse grafisch einfacher zu gestalten, indem durch das mehrmalige Auftreten komplizierte Daten-Input/Outputs gespart werden können.

Der Lebenszyklus eines `Data Objects` entspricht dem des Eltern `Process` bzw. `Sub-Process`. Sobald der `Process` bzw. `Sub-Process` instanziiert wird, wird auch das `Data Object` instanziiert, wird der `Process` bzw. `Sub-Process` beendet, wird das `Data Object` ebenfalls beendet. Die Daten sind nicht mehr verfügbar. Die Sichtbarkeit ist an diesen Lebenszyklus gekoppelt. Eine `Data Object` ist nur für den direkten übergeordneten `Process` bzw. `Sub-Process` und alle Geschwisterelementen im gleichen Diagramm, sowie deren Kindern sichtbar.

Daten, die durch ein `Data Object` modelliert werden, sind lediglich zur Prozessausführung vorhanden. Um Daten, die über die Laufzeit des Prozesses gültig sind, zu verwalten werden `Data Stores` (siehe Abb. 3.2e) verwendet.

`Properties` sind wie auch `Data Objects` `Item-Aware Elements`. Aber im Gegensatz dazu sind sie in einem Prozessdiagramm nicht visualisiert. `Processes` und `Activities` können `Properties` enthalten. Die `Property` dient dabei als Container für Daten des jeweiligen Elements. Der Lebenszyklus einer `Property` ist verbunden mit dem Lebenszyklus des Elements, in dem sie enthalten ist. Die Sichtbarkeit wird wiederum vom Lebenszyklus bestimmt. Auf eine `Property` kann nur von seinem umgebenden Element zugegriffen werden, ist dies ein `Process` oder `Sub-Process`, dann auch von dessen Kinder.

Für die Ausführungen benötigen `Processes` und `Activities` oft Daten. Zudem werden während der Ausführung und als Resultat meist Daten generiert. Daten, die benötigt werden, werden in `Data Inputs` und `Input Sets` beschrieben, Daten, die generiert werden, in `Data Outputs` und `Output Sets`. Aggregiert werden diese Elemente in einer `InputOutputSpecification`, die dann beispielsweise einer `Activity` angehängt wird, um deren Datenanforderungen zu beschreiben.

`Data Inputs` sind wie `Data Objects` `Item-Aware Elements`. Sie haben die gleiche grafische Darstellung, enthalten aber noch einen nicht ausgefüllten Pfeil und keinerlei eintretende Kanten (siehe Abb. 3.2c). Sie repräsentieren die Informationen, die benötigt werden um eine `Activity` zu starten. Es ist möglich sie als optional oder als erforderlich zu deklarieren. Wird der `Data Input` als erforderlich deklariert, kann die verbundene `Activity` erst starten, wenn die Informationen

vorliegen. `Data Outputs` haben dieselbe Darstellung wie `Data Inputs`, jedoch mit einem ausgefüllten Pfeil (siehe Abb. 3.2d). Sie haben keine ausgehenden Kanten. Oft werden `Data Input` und `Data Output` nicht in einem Diagramm visualisiert. Sie sind dann ausschließlich im zugrunde liegenden XML-Dokument spezifiziert (siehe Listing 5.2, vgl. dazu Abb. 4.6).

Eine Kollektion von `Data Inputs` (Zeile 21-24) wird in einem `Input Set` (Zeile 27-31) zusammengefasst, eine Kollektion von `Data Outputs` (Zeile 25) in einem `Output Set` (Zeile 32-35). Diese wiederum bilden eine `InputOutputSpecification` (Zeile 20-36). Es ist auch möglich in einer `InputOutputSpecification` mehrere `Input Sets` zu definieren. In diesem Fall wird der Reihe nach ausgewertet und das erste Vollständige ausgewählt.

Eine `Data Association`, die im Diagramm optional dargestellt werden kann (siehe Abb. 3.3c), ist in einer `Activity` oder einem `Event` enthalten. Sie wird dazu benutzt, Daten aus `Item Aware Elements` zu lesen bzw. in sie zu schreiben. Sie hat jeweils mehrere Quellen und ein Ziel, wobei bei der Ausführung die Quellen ins Ziel kopiert werden. Die Quelle und das Ziel müssen dabei auf die gleiche `Item Definition` verweisen. Ist dies nicht der Fall, muss eine Transfomation mit Hilfe einer `Expression` durchgeführt werden. Die `Data Association` kann weiter unterteilt werden in eine `Data Input Association`, die ein `Item-Aware Element` mit einer `Activity` verbindet, und eine `Data Output Association`, die eine `Activity` mit einem `Item-Aware Element` verbindet.

Um die modellierten Daten abfragen zu können, um beispielsweise bei einem Gateway festzustellen, welche ausgehende Kante aktiviert werden muss, können `Formal Expressions` verwendet werden, welche die `Expressions` erweitern und eine formale Definition ermöglichen. `Formal Expressions` greifen auf eine Abfragesprache zurück, die wie auch die Strukturbeschreibungssprache global für den Prozess definiert wird, jedoch für einzelne `Formal Expressions` überschrieben werden kann. Die Abfragesprache muss jeweils zu der verwendeten Strukturbeschreibungssprache passen. Standardmäßig wird in der BPMN die zu XML-Schema passende Abfragesprache XPath verwendet. Listing 5.3 zeigt die Verwendung einer `Formal Expression` innerhalb eines `Sequence Flows`. Die Kante wird aktiviert, wenn im Kundenobjekt das Attribut "akzeptiert" gesetzt ist.

5.3 Ausnahmebehandlung

5.3.1 Sicht der Systemmodellierer

Die im fachlichen Modell erkannten und textuell beschriebenen Ausnahmen müssen nun formal in den Prozessablauf eingebettet werden. Hierzu gehört insbesondere die detaillierte Beschreibung, was eine Ausnahme auslöst und was danach geschieht.

Da es sich bei Ausnahmen um Vorkommnisse handelt, die den Prozessablauf direkt beeinflussen, ist es sinnvoll, diese grafisch zu modellieren. Dies hat auch den Vorteil, dass die Ausnahmen und die darauf folgende Reaktion im Gegensatz zu einer rein textuellen Beschreibung für den Betrachter eines Modells leichter erfassbar sind. Um die Ausnahmen grafisch beschreiben zu können, können wie in Kapitel 4.3.1 bereits erläutert, verschiedenste grafische Elemente eingeführt werden. Um beispielsweise zu modellieren, dass ein Prozessschritt innerhalb von maximal 60 Minuten abgearbeitet sein muss, kann das Symbol einer Uhr auf einem gestrichelten Pfeil verwendet

```
<definitions ...
             typeLanguage="http://www.w3.org/2001/XMLSchema"
             xmlns:sd="http://www.daimler.com/sample">
...
<import importType="http://www.w3.org/2001/XMLSchema"
        location="sample.xsd"
        namespace="http://www.daimler.com/sample">
<itemDefinition id="kundeStruktur" structure="sd:kunde"/>
<itemDefinition id="vertragStruktur" structure="sd:vertrag"/>
<itemDefinition id="rechnungStruktur" structure="sd:rechnung"/>
...
<process id="abrechnung" name="Abrechnung">
  <dataObject id="kunde" name="Kunde" itemSubjectRef="kundeStruktur"/>
  <dataObject id="vertrag" name="Vertrag" ↩
  itemSubjectRef="vertragStruktur"/>
  <dataObject id="rechnung" name="Rechnung" ↩
  itemSubjectRef="rechnungStruktur"/>
  ...
  <userTask id="kostenBerechnen" name="Kosten berechnen">
    <ioSpecification>
      <dataInput id="berechne-input1" structureDefinitionRef="sd:kunde" ↩
      optional="false"/>
      <dataInput id="berechne-input2" structureDefinitionRef="sd: ↩
      vertrag" optional="false"/>
      <dataOutput id="berechne-output" structureDefinitionRef="sd: ↩
      rechnung"/>
      <inputSet id="berechne-is">
        <dataInputRefs>berechne-input1</dataInputRefs>
        <dataInputRefs>berechne-input2</dataInputRefs>
        <outputSetRefs>berechne-os</outputSetRefs>
      </inputSet>
      <outputSet id="berechne-os">
        <dataOutputRefs>berechne-output</dataOutputRefs>
        <inputSetRefs>berechne-is</inputSetRefs>
      </outputSet>
    </ioSpecification>
    ...
  </userTask>
  ...
</process>
</definitions>
```

Listing 5.2: Definition der Eingabe- und Ausgabedaten einer Activity

```
01 <sequenceFlow id="flow1" sourceRef="kundeUeberpruefen" ↩
02 targetRef="FahrzeugAuswaehlen">
03   <conditionExpression xsi:type="tFormalExpression">
04     getDataObject('kunde')/akzeptiert=true
05   </conditionExpression>
06 </sequenceFlow>
```

Listing 5.3: Formal Expression innerhalb eines Sequence Flows

werden (siehe Abb. 5.1). Die entsprechende Reaktion auf die Ausnahme wäre in diesem Fall, dass der Vorgesetzte über die Verzögerung informiert wird.

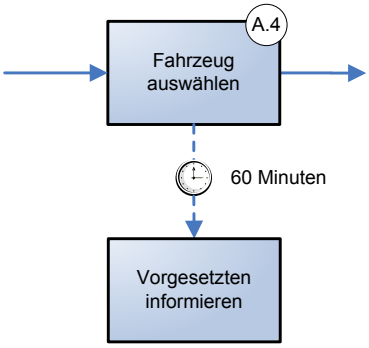


Abbildung 5.1: Grafische Modellierung von Ausnahmen

Über die grafische Darstellung hinaus, muss jedoch noch mehr Information hinterlegt werden, um den Geschäftsprozess ausführbar zu machen. Beispielsweise muss die Zeitspanne, nach der im obigen Beispiel der alternative Pfad ausgeführt werden soll, formal spezifiziert werden. Hierzu müssen geeignete Möglichkeiten geschaffen werden, um diese Informationen in Attributen der jeweiligen Elemente angeben zu können.

Wie bereits besprochen, kann das Eintreten einer Ausnahme unterschiedliche Folgen nach sich ziehen. Es ist denkbar, dass lediglich ein paralleler Pfad gestartet wird, möglich ist aber auch, dass bereits ausgeführte Prozessschritte rückgängig gemacht werden müssen. Aus diesem Grund ist es wünschenswert, Geschäftsprozesse transaktional nach dem ACID[1]-Paradigma [LÖ09] ausführen zu können. Das heißt, die einzelnen Prozessschritte sollen entweder alle oder gar nicht durchgeführt werden. Da Geschäftsprozesse jedoch langläufig sind und in ihrem Ablauf menschliche Interaktionen, asynchrone Serviceaufrufe und Wartezustände vorkommen, ist dies schwer zu realisieren. Im Umfeld des BPM gibt es deshalb den Begriff der fachlichen Transaktion [Pap03]. Einzelne Serviceaufrufe können jederzeit als technische Transaktion, die dem ACID-Paradigma folgen, realisiert werden. Ist ein solcher Serviceaufruf jedoch einmal abgeschlossen, kann er im technischen Sinne nicht mehr rückgängig gemacht werden. Fachlich ist dies jedoch durch das Ausführen eines zusätzlichen Kompensationsschrittes möglich. Im Prozessschritt 1.4 "Fahrzeug auswählen" wurde zum Beispiel bereits ein Fahrzeug ausgewählt. Aufgrund einer Stornierung

[1] ACID steht für atomicity, consistency, isolation und durability. Auf deutsch atomar, konsistent, isoliert und dauerhaft

muss diese Auswahl jedoch rückgängig gemacht werden. Dies kann durch das Einfügen eines Kompensationsschrittes "Fahrzeugauswahl rückgängig machen" realisiert werden. Problematisch ist jedoch, dass nicht für jeden Service Kompensationsservices angeboten werden. Ein bereits gebuchter Billigflug kann zum Beispiel 1 Woche vor dem Start nicht mehr kostenfrei storniert werden. Zur Beschreibung transaktionaler Aspekte in Geschäftsprozessen wurde mit WS-Transaction [OAS05] bereits ein geeigneter Standard definiert. In diesem werden Protokolle sowohl für kurzlebige Aktivitäten, die ganz oder gar nicht ausgeführt werden können, als auch für langlaufende Transaktionen, für die die Möglichkeit der Kompensation geschaffen werden muss, definiert.

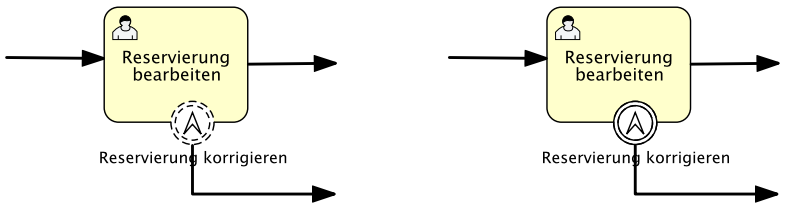
Ausnahmen sollten grafisch modelliert werden können. Dabei muss der Gültigkeitsbereich der Ausnahmen deutlich werden. Um technische Details, die zur Ausführung des Geschäftsprozesses benötigt werden, in das Modell integrieren zu können, sollten zudem passende Attribute vorhanden sein. Darüber hinaus müssen Transaktionen realisierbar sein.

5.3.2 Umsetzung in BPMN

Die im fachlichen Modell identifizierten und dort als Text hinterlegten Ausnahmen werden nun im Modell visualisiert und im zugrunde liegenden XML weiter formalisiert. Die in Kapitel 4.3.1 definierten Ausnahmen können wie folgt in BPMN umgesetzt werden.

Interne Ausnahmen im Prozess:
Können wie bereits in Kapitel 4.3.2 beschrieben durch Gateways realisiert werden.

Ausnahmen bei Benutzerinteraktion:
Können durch `Escalation-Events`, die als `Boundary Events` an der entsprechenden `Activity` angedockt werden, modelliert werden. Dabei gibt es eine nicht unterbrechende Variante (siehe Abb. 5.2a), bei der ein paralleler Sequenzfluss gestartet wird und eine unterbrechende Variante (siehe Abb. 5.2b), bei der die Ausführung der `Activity` abgebrochen und ein alternativer Sequenzfluss gestartet wird. Eine Möglichkeit eine `Activity` für die Ausnahmebehandlung anzuhalten und später fortzuführen existiert nicht.



(a) nicht unterbrechend (b) unterbrechend

Abbildung 5.2: Modellierung einer Ausnahme bei Benutzerinteraktion

Fehler:
Können wie Ausnahmen bei Benutzerinteraktionen durch `Boundary Events` modelliert werden. Hier wird jedoch der `Error-Event` verwendet. Eine nicht unterbrechende Variante macht an

dieser Stelle keinen Sinn, weshalb es in der BPMN auch kein nicht unterbrechendes `Error-Event` gibt. Abb. 5.3 zeigt einen Ausschnitt aus dem Reservierungsprozess. Der Serviceaufruf "Kunde überprüfen" kann aufgrund fehlender oder falscher Daten einen Fehler verursachen. Dieser Fehler wird abgefangen und die Daten werden korrigiert, bevor ein neuer Versucht gestartet wird.

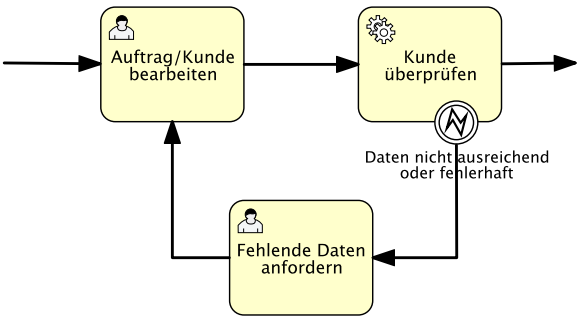


Abbildung 5.3: Fehler bei Ausführung eines automatischen Prozessschrittes

Deadlines:
Können mit Hilfe des `Timer-Events` modelliert werden. Durch die Angabe einer Zeitspanne nach Beginn (z.B. 60 Minuten) oder eines bestimmten Zeitpunktes (z.B. 12:00 Uhr) kann festgelegt werden, wann die Deadline ausgelöst wird. Es gibt jedoch keine Möglichkeit zu unterscheiden, wie weit der Prozessschritt bereits abgearbeitet wurde, um zum Beispiel nur zu unterbrechen, wenn weniger als 50% bearbeitet wurden. Zudem kann eine Deadline nur direkt hintereinander ablaufende Schritte betrachten. Es ist also nicht möglich, einen Zwischenschritt von der Zeitmessung auszuschließen oder die Messung anderweitig zu unterbrechen. Will man solche Konstrukte modellieren, kann als Workaround ein Datenobjekt verwendet werden, in dem ein Zeitstempel gespeichert wird, der von verschiedenen Stellen innerhalb des Prozessablaufs gelesen und geschrieben werden kann.

Unerwartete externe Ausnahmen:
Eingehende Nachrichten, wie zum Beispiel eine Stornierung, können einzelne `Activities` oder komplette Prozesse beenden. Mit Hilfe des `Message-Events` können die eintreffenden Nachrichten abgefangen werden und je nach Bedarf ein paralleler oder alternativer Ausführungspfad gewählt werden. Abbildung 5.4 zeigt den Hauptprozess der Autovermietung des Anwendungsszenarios. Der `Sub-Process` "Reservierung", wird abgebrochen, sobald der Kunde einen Stornierungswunsch schickt. Damit wurde der normale Kontrollfluss gestoppt. Es wird nun lediglich noch der `Sub-Process` "Auftrag stornieren" ausgeführt, bevor der Prozess mit dem Senden einer Stornierungsbestätigung an den Kunden beendet wird.

Ausnahmen bei angeforderten Nachrichten:
Verschickt man an einen Geschäftspartner eine Anfrage, rechnet man mit einer bestimmten Antwort. Diese fällt jedoch nicht immer wie erwünscht aus. Um die verschiedenen potentiellen Antwortmöglichkeiten modellieren zu können, exisitert in der BPMN das `Event-based Gateway`. Dieses bietet, wie in Abb. 5.5 dargestellt die Möglichkeit, die verschiedenen Antwortmöglichkeiten und die darauf passende Reaktion zu modellieren.

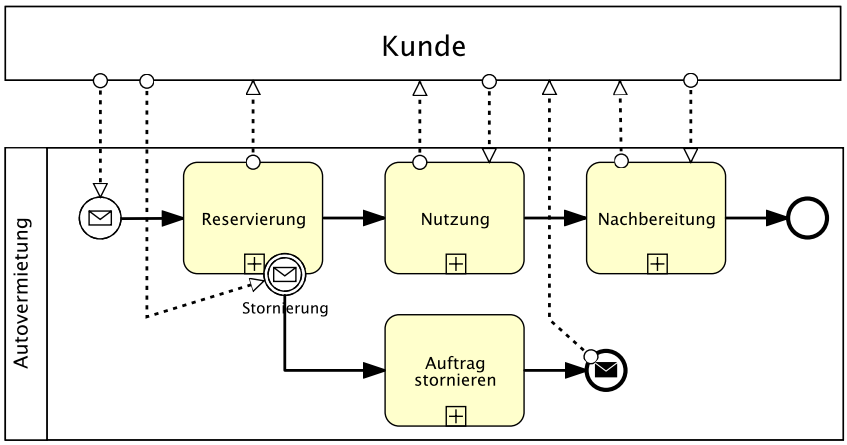


Abbildung 5.4: Unerwartete externe Nachricht führt zum Stornieren

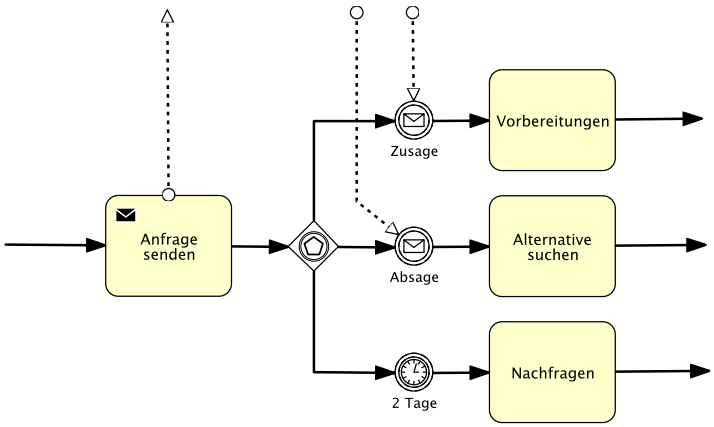


Abbildung 5.5: Unerwartete externe Nachrichten nach einer Anfrage

Neben der Möglichkeit `Events` als `Boundary-Events` zu modellieren, kann auch ein sogenannter `Event-Subprocess` (siehe Abb. 3.8c) verwendet werden, um Ausnahmen abzufangen. Er ist Teil eines Prozesses und wird demzufolge auch in dessen Kontext ausgeführt. Er hat keine eingehenden und ausgehenden Kanten und wird aktiviert, sobald das in ihm modellierte Startereignis eintritt. Der umgebende Prozess kann durch den `Event-Subprocess` beendet werden (unterbrechend), oder der `Event-Subprocess` wird parallel zum umgebenden Prozess ausgeführt (nicht unterbrechend). Der umgebende Prozess wird beendet, sobald sowohl der eigentliche Prozess, als auch alle `Event-Subprocesse` beendet wurden. Aus diesem Grund bietet sich diese Möglichkeit, insbesondere für kurze Ausnahmebehandlungen an. Bei längeren Ausnahmebehandlungen ist es eventuell sinnvoller `Boundary-Events` zu verwenden. Dauert eine nicht unterbrechende Ausnahmebehandlung in einem `Event-Subprocess` zum Beispiel deutlich länger als der übergeordnete

Prozess, ist es sinnvoller diesen mit Hilfe eines `Boundary-Events` zu modellieren, um den eigentlichen Prozess weiter ablaufen zu lassen. Voraussetzung ist allerdings, dass auch in diesem Kontext auf alle relevanten Objekte zugegriffen werden kann. Abb. 5.6 zeigt ein Beispiel eines `Event-Subprocess`. Tritt während der Ausführung des "Subprozess" die definierte Ausnahme ein, wird die Ausführung beendet und der im `Event-Subprocess` definierte Prozess ausgeführt. Anschließend wird mit dem Sequenzfluss nach "Subprozess" fortgefahren.

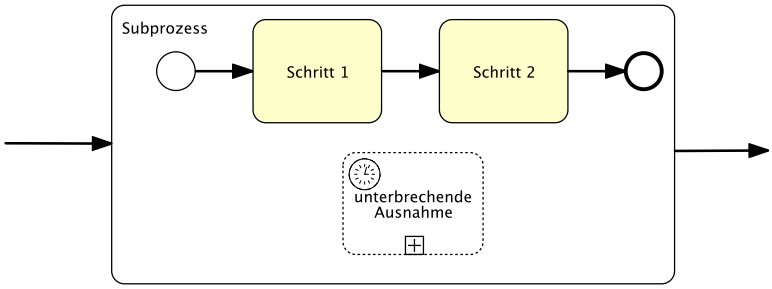


Abbildung 5.6: Event-Subprocess innerhalb eines Subprocesses

Wie bereits beschrieben lassen sich Ausnahmen nur für einzelne `Activities` oder `Sub-Processes` realisieren. Soll eine Ausnahme über mehrere Prozessschritte, jedoch nicht den ganzen Prozess gültig sein, wie im Anwendungsszenario die Stornierung, kann dieser Teil in einen eigenen `Sub-Process` eingefasst werden. Durch ein `Boundary-Event` oder einen `Event-Subprocess` lässt sich dann die Ausnahme modellieren. Der Nachteil ist jedoch, dass der Prozess durch das Einfügen eines zusätzlichen `Sub-Processes` unter Umständen unübersichtlicher wird.

Neben der visuellen Modellierung der Ausnahmen, müssen die formalen Details im zugrunde liegenden XML erfasst werden. Dies ist in der BPMN möglich, jedoch teilweise mit Einschränkungen verbunden. Bei einem `Timer-Event` muss beispielsweise formal angegeben werden können, zu welchem Datum bzw. Uhrzeit oder nach welcher Zeitspanne es ausgelöst wird. Die Angabe ist jedoch nur durch eine `Expression` möglich, die nichts anderes als ein reiner Freitext ist. Hier sind wiederum die Hersteller der Process-Engines gefragt, die sich hier selbst eine eigene Semantik definieren müssen, an welche sich der Modellierer halten muss.

Erkannte Ausnahmen können sofort behandelt werden. Es ist aber auch möglich, erkannte Ausnahmen weiter zu propagieren, um sie erst auf einer höheren Ebene zu behandeln. Beim Auswählen eines Fahrzeuges wird zum Beispiel erkannt, dass kein Fahrzeug vorhanden ist (siehe Abb. 5.7). Dies führt dazu, dass der `Subprocess` "Reservierung" den '`Error-Event` "kein Fahrzeug vorhanden" wirft. Dieser wird wiederum abgefangen und führt dazu, dass der Auftrag durch das Senden einer Nachricht abgelehnt wird.

In der BPMN 2.0 wurden Transaktionen neu eingeführt. Damit kann definiert werden, was bei bereits abgeschlossenen `Activities` im Fehlerfall getan werden soll, um diese fachlich zurückzusetzen. Hierzu wird ein `Transaction-Subprocess` verwendet. Die innerhalb dieser Transaktion befindlichen Schritte sollen entweder alle oder keiner davon ausgeführt werden. Dazu müssen im Fehlerfall, der durch ein `Cancel-Event` deutlich gemacht wird, Schritte eingefügt werden, die

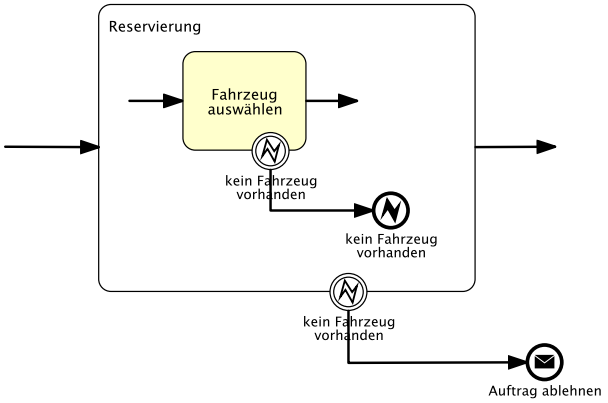


Abbildung 5.7: Erkannter Fehler wird weiter propagiert

bereits durchgeführte Änderungen fachlich rückgängig machen. Hierzu können für die einzelnen `Activities` mit Hilfe von `Compensation-Events` `Compensation-Activities` definiert werden, die im Fehlerfall aufgerufen werden. Abb. 5.8 macht die Verwendung von Transaktionen deutlich. Nachdem das Fahrzeug reserviert wurde, soll ein Mietvertrag ausgestellt werden. Dies kann jedoch, aus welchen Gründen auch immer, nicht ausgeführt werden und es wird ein Fehler geworfen. Dieser Fehler wird durch ein `Boundary-Event` abgefangen, welches wiederum ein `Cancel-Event` initiiert. Durch den Abbruch des `Transaction-Subprocesses` werden alle bereits ausgeführten `Activities`, in diesem Fall die Reservierung des Fahrzeuges, durch `Compensation-Activities` zurückgesetzt. Anschließend wird der Kunde noch durch eine Nachricht informiert. Damit ist gewährleistet, dass die Transaktion entweder als Ganzes ausgeführt wird oder im Falle eines Fehlers der Ursprungszustand wieder hergestellt wird.

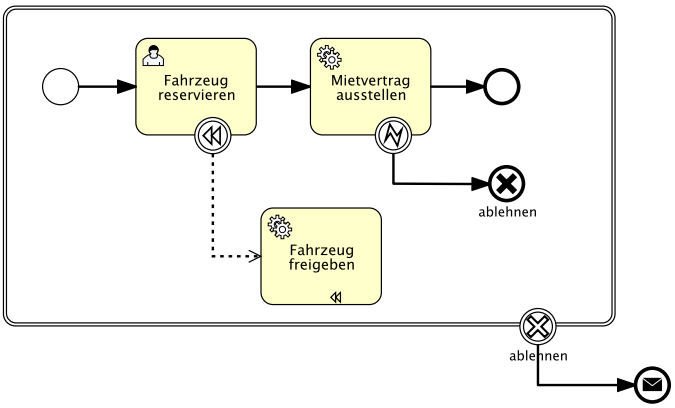


Abbildung 5.8: Transaktionen im Geschäftsprozess

Die im fachlichen Modell als strukturierter Text definierten Ausnahmen können mit den be-

schriebenen Elementen im Diagramm umgesetzt werden. Die Vielzahl an Events in der BPMN ermöglichen eine gute Ausnahmebehandlung sowohl von fachlichen Ausnahmen als auch von technischen Fehlern. Bei der formalen Detaillierung zeigt die BPMN jedoch teilweise Schwachstellen.

5.4 Flexibilitätsstellen

5.4.1 Sicht der Systemmodellierer

Für die im Fachmodell definierten Flexibilitätsstellen muss nun eine geeignete flexible Implementierung realisiert werden. Die unterschiedlichen Typen von Flexibilität werden dabei unterschiedlich behandelt.

Datenflexibilität:
Werden Daten geändert, kann dies unter Umständen einen Einfluss auf den Ablauf des Prozesses haben. Dies ist dann der Fall, wenn bei Verzweigungen auf Daten zurückgegriffen wird und diese die Entscheidung beeinflussen. Werden Kunden unter 30 Jahren zum Beispiel anders behandelt als Kunden über 30 Jahren (ihnen wird beispielsweise noch eine Zusatzversicherung angeboten), kann das Attribut "Alter" des Kunden nicht einfach gelöscht werden. Aus diesem Grund sind Änderungen an der Datenstruktur immer kritisch und es liegt in der Verantwortung des Systemmodellierers, solche Fälle auszuschließen. Eine einfache Entkopplung der Daten von der Prozesslogik ist nur dann möglich, wenn es sich nicht um prozessrelevante Daten handelt.

Bearbeiterflexibilität:
Eine Veränderung in der Organisationsstruktur muss nicht notwendigerweise den Prozessablauf beeinflussen, da diese ausgelagert ist. Eine Änderung kann daher jederzeit durchgeführt werden ohne eine Beeinflussung der Prozesslogik. Werden zum Beispiel neue Mitarbeiter eingestellt oder die Kompetenzen eines bestehenden Mitarbeiters geändert, muss der Prozess nicht angepasst werden. Werden jedoch ganze Organisationseinheiten oder Gruppen gelöscht, müssen unter Umständen die Zuordnungsregeln der Bearbeiter zu den Prozessschritten (vgl. Kapitel 4.1) ebenfalls angepasst werden [RCWR04].
Die Bearbeiterzuordnung durch die Regeln zu den einzelnen Prozessschritten kann hart kodiert oder in eine eigene Engine ausgelagert werden. Wird die Zuordnung hart kodiert, muss bei einer Änderung der Regeln der Prozess angepasst und in die Process-Engine neu "deployed" werden. Durch das Auslagern in eine eigene Engine müssen lediglich dort die Regeln geändert werden. Der Aufruf der Engine im Prozessablauf wird nicht beeinflusst. Auf Änderungen der Regeln bei der Bearbeiterzuordnung kann so flexibler reagiert werden.

Flexibilität an Verzweigungsbedingung:
Mit Hilfe von Verzweigungen und insbesondere durch Hintereinanderschaltung von mehreren Verzweigungen, können komplexe Regeln modelliert werden. Der Prozessschritt A.2 "Kunde überprüfen" kann zum Beispiel weiter detailliert werden (siehe Abb. 5.9). Ein Teil der weiteren Detaillierung ist, dass die Bonität des Kunden durch eine Schufaanfrage geprüft wird. Diese wird jedoch nur durchgeführt, wenn der Kunde ein Neukunde ist. Ist der Kunde kein Neukunde, aber übersteigt der Auftragswert 1.000 Euro wird trotzdem eine Überprüfung durchgeführt. Hat der Kunde allerdings bereits einen Gesamtumsatz von mehr als 10.000 Euro, muss die Bonität nicht

geprüft werden.
Eine Modellierung dieses Sachverhaltes kann wie in Abbildung 5.9 dargestellt über Verzweigungen innerhalb des Prozessmodells realisiert werden. Die grafische Modellierung dieser Regeln kann jedoch sehr schnell unübersichtlich werden. Änderungen können nur schwer eingearbeitet und in laufende Prozessinstanzen propagiert werden. Zudem sind die definierten Regeln an anderen Stellen nicht wiederverwendbar.

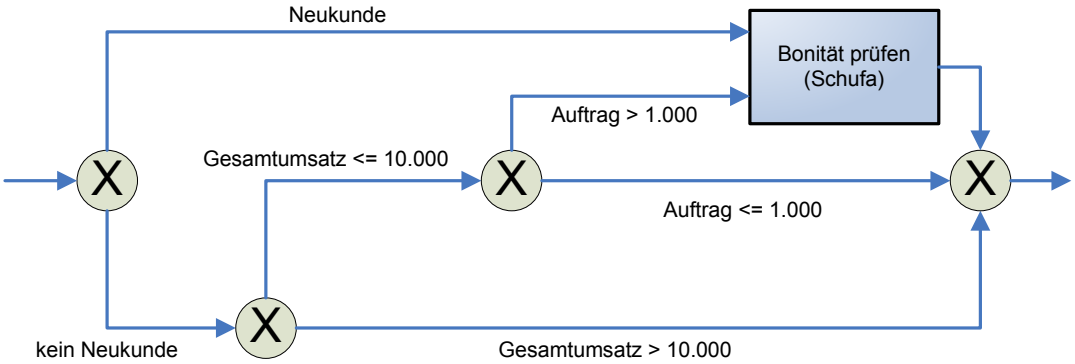


Abbildung 5.9: Komplexe Verzweigungen in einem Geschäftsprozess

Durch eine Auslagerung der Regeln in ein so genanntes Business-Rule-Management-System (BRMS) kann die Prozesslogik von der Geschäftslogik getrennt werden. In einem BRMS können Regeln verwaltet werden. Durch einen Aufruf, bei dem die benötigten Daten übergeben werden, wird die Regel angewendet und das Ergebnis zurückgegeben. Durch diese Trennung der Prozesslogik von der Geschäftslogik wird der modellierte Prozess übersichtlicher und einfacher wartbar. Wird zum Beispiel die Schranke für die Bonitätsprüfung eines Kunden von 1.000 Euro auf 2.000 Euro erhöht, bleibt das Prozessmodell unangetastet. Lediglich im BRMS muss die Regel angepasst werden. Abbildung 5.10 zeigt den Prozess aus Abbildung 5.9. Allerdings wurde die Geschäftslogik, die entscheidet, ob ein Kunde überprüft wird oder nicht, in ein BRMS ausgelagert. Im Prozessmodell wird lediglich noch eine Verzweigung benötigt, die den weiteren Sequenzfluss aufgrund der getroffenen Entscheidung regelt.

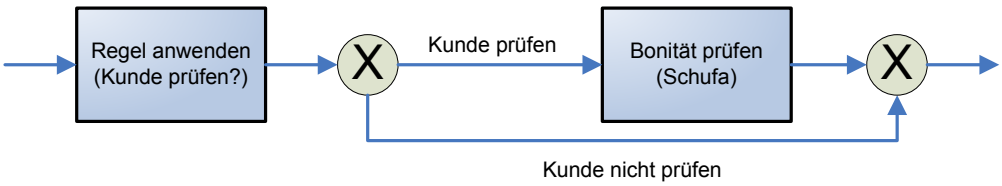


Abbildung 5.10: Regeln ausgelagert in BRMS

Prozessschrittflexibilität:
Wurde im Fachmodell ein Prozessschritt als Flexibiliätsstelle markiert, bietet sich im Systemmodell unter Umständen eine Aufspaltung in einen flexiblen und einen unflexiblen Teil an. Diese Aufspaltung kann jedoch nur durchgeführt werden, wenn die beiden Teile nicht voneinander abhängig sind, sondern klar getrennt werden können. Der flexible Teil kann, wenn es sich lediglich um Regeln handelt, wiederum durch den Aufruf eines BRMS realisiert werden.

Flexibilität bei der Servicezuweisung:
Flexibilität bei der Servicezuweisung kann erreicht werden, indem der Service nicht direkt gerufen wird. Stattdessen wird der Aufruf entkoppelt über einen Proxy (bspw. Enterprise Service Bus [TNB06]) durchgeführt, der den eigentlichen Serviceaufruf realisiert. Dies führt dazu, dass der Prozessablauf bei einer Änderung des Serviceaufrufs stabil bleibt. Es muss lediglich der Proxy angepasst werden, der Prozess muss nicht neu "deployed" werden.
Zusätzlich zum Proxy kann ein Repository eingesetzt werden. Der Serviceaufruf muss so nicht hart im Proxy-Ablauf kodiert werden, sondern kann dynamisch zur Laufzeit ermittelt werden. Bei einer Änderung des Serivceaufrufs muss lediglich der Eintrag im Repository geändert werden. Weder der Prozess noch der Proxy-Ablauf müssen bei einer Änderung neu "deployed" werden.

5.4.2 Umsetzung in BPMN

Um Flexibilität an Verzweigungsbedingung zu modellieren, existiert ein eigenes Element um Geschäftsregeln aufrufen zu können, die extern in einem BRMS verwaltet werden. Der `Business Rule Task` übernimmt diese Aufgabe (vgl. Abschnitt 3.2.2). Der `Business Rule Task` muss lediglich mit den nötigen Eingabedaten versorgt werden und liefert das Ergebnis der Berechnung zurück. Nach der Berechnung der Regeln wird ein `Gateway` zur Auswahl des richtigen Weges modelliert. Anhand der zurückgegebenen Informationen des `Business Rule Tasks` wird die Entscheidung über den weiteren Verlauf des Sequenzflusses vorgenommen (siehe Abb. 5.11).

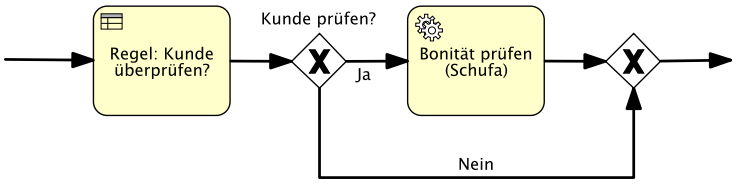


Abbildung 5.11: Verwendung einer Business Rule Task

Um die weiteren Konzepte in BPMN umzusetzen, bedarf es keiner neuen Elemente. Stattdessen sind es lediglich Fragen des Designs, die eine Flexibilität ermöglichen. Wurde etwa an einen Prozessschritt eine Flexibilität bei der Servicezuweisung markiert, wird der Service nicht direkt aufgerufen. Stattdessen wird der Aufruf durch einen Proxy entkoppelt, sodass bei einer Migration des Services lediglich der Ablauf des Proxys angepasst werden muss. Ähnliche Konzepte können auch bei den anderen Typen von Flexibilität angewendet werden.

5.5 Zusammenfassung

In diesem Kapitel wurden Konzepte vorgestellt um die Datenmodellierung, die Bearbeiterzuordnung, die Ausnahmebehandlung und Flexibilitätsstellen formal im Systemmodell umzusetzen. Dazu wurde auf das von der Fachabteilung modellierte fachliche Modell aufgebaut. Dieses wurde formalisiert und unter Umständen durch Rückfragen bei der Fachabteilung vervollständigt.

Das so entstandene Systemmodell setzt nun alle Anforderungen der Fachabteilung vollständig um. Eine weitere Rücksprache mit der Fachabteilung ist nicht erforderlich. Zudem ist es vollständig formalisiert. Das entstandene Systemmodell dient nun als Grundlage für das ausführbare Modell.

6

Weiterentwicklung zur fertigen Applikation

Im Systemmodell sind nun alle Anforderungen der Fachabteilung an die zu entwickelnde Applikation spezifiziert. Das Modell ist formal beschrieben und lässt keine weiteren Interpretationsspielräume zu. Es kann folglich von einem Computer interpretiert werden. Im nächsten Schritt muss das Systemmodell in ein ausführbares Modell überführt werden, die Software-Realisierung des modellierten Geschäftsprozesses. Da das ausführbare Modell nicht der Schwerpunkt dieser Arbeit ist, werden lediglich die Vorgehensweise und die verwendeten Sprachen skizziert.

In Abschnitt 6.1 wird die grundlegende Überführung des Systemmodells in ein ausführbares Modell beschrieben. Anschließend wird dies mittels BPMN realisiert.

6.1 Allgemeine Umsetzung

Das ausführbare Modell wird von IT-Implementierern erstellt. Diese treffen keine (fachliche relevanten) Entscheidungen bei der Erstellung des Modells, sondern übernehmen die Struktur und die Inhalte des Systemmodells 1:1.

Um in einer serviceorientierten Architektur Geschäftsprozesse durch Unterstützung von serviceorientierter Workflow-Technologie [Rei00, Wes07, Jab97] zu realisieren, muss das Systemmodell in eine dafür geeignete Ausführungssprache überführt werden. Häufig findet dabei BPEL Verwendung, eine XML-basierte Sprache [OAS07]. Darüber hinaus existieren weitere Ausführungssprachen [KLL09], wie zum Beispiel die XML Process Definition Language (XPDL) [WFM02] oder die BPML [Sha02]. Alternativ zur Verwendung von Workflow-Technologie, kann der Geschäftsprozess auch hart kodiert und etwa als J2EE oder .NET Anwendung realisiert werden. Dies ist insbesondere bei kleineren Projekten interessant, da so hohe Lizenzkosten und der Aufwand für die Mitarbeiterschulung gespart werden können.

In einer serviceorientierten Architektur werden die Services in eine Ablaufreihenfolge gebracht. Dies wird mit Hilfe einer Ausführungssprache durchgeführt und wird als Orchestrierung [Erl05] bezeichnet. Der genaue Ablauf wird dabei vom Systemmodell vorgegeben und 1:1 übernommen. Die im Systemmodell verwendete Modellierungssprache (UML, EPK, BPMN, etc.) muss folglich in eine Ausführungssprache (BPEL, XPDL, BPML, etc.) überführt werden. So ist ein Systemmodell-Prozess zum Beispiel als UML Aktivitätsdiagramm modelliert und soll als BPEL-Prozess implementiert werden. Dazu werden die im Systemmodell dokumentierten manuellen Prozessschritte etwa als Human Tasks [Agr07] realisiert. Außerdem müssen geeignete BPEL-Konstrukte (z.B. While, Parallel-ForEach) verwendet werden, um den im Systemmodell dokumentierten Kontrollfluss in einem BPEL-Prozess zu realisieren. Diese Überführung vom Systemmodell ins ausführbare Modell kann mit gewissen Einschränkungen automatisiert werden [ZD08, ODBH05, SI07].

Um den in einer Ablaufsprache definierten Geschäftsprozess ausführen zu können, wird eine entsprechende Ausführungsumgebung benötigt, eine so genannte Process-Engine (z.B. IBM WebSphere Process Server [IBM08]). Der modellierte Prozess muss dazu in die Process-Engine deployed werden.

6.2 Umsetzung mit BPMN

In der Spezifikation der BPMN 2.0 ist eine eigene Ausführungssemantik für die BPMN enthalten. Somit können Modelle, die in BPMN modelliert wurden, durch eine BPMN 2.0 kompatible Process-Engine direkt ausgeführt werden. Eine Überführung in eine Ausführunssprache wie zum Beispiel BPEL ist nicht erforderlich. Durch die Verwendung der BPMN auf der Ebenen des ausführbaren Modells kann eine durchgängige Modellierung vom fachlichen Modell, über das Systemmodell bis hin zum ausführbaren Modell theoretisch realisiert werden.

Alternativ können die mit der BPMN modellierten Geschäftsprozesse in eine andere Ausführungssprache überführt werden. Der Überführung in die bekannteste Ausführungssprache BPEL ist in der Betaversion der BPMN 2.0 Spezifikation ein eigenes Kapitel gewidmet. Da BPEL im Gegensatz zu der graphbasierten BPMN blockstrukturiert ist, ist die Überführung jedoch keinesfalls trivial [ODHA06].

Im Sinne des Business-IT-Alignments ist eine direkte Ausführung von BPMN wünschenswert. Durch die direkte Ausführung können zudem Probleme umgangen werden, die durch die Überführung in BPEL entstehen. Für eine direkte Ausführung der BPMN wird jedoch eine geeignete Process-Engine benötigt.

Im nächsten Kapitel werden drei Werkzeuge zur Modellierung von Geschäftsprozessen vorgestellt und bewertet. Davon bietet jedoch lediglich eines eine eigene Process-Engine, um die modellierten Prozesse ausführen zu können. In den anderen Werkzeugen muss das entstandene Modell exportiert und in einer anderen Process-Engine ausgeführt werden.

7 Werkzeugunterstützung

Um BPMN Diagramme modellieren zu können, wird ein passendes Werkzeug benötigt. Auf dem Markt befinden sich viele Werkzeuge zur Modellierung von BPMN Diagrammen in der Version 1.0. Dies reicht von kostenlosen Programmen zur reinen Modellierung bis hin zu mehreren tausend Euro teuren Programmen, die eine integrierte Process-Engine besitzen. Nachdem im August 2009 die Betaversion der BPMN 2.0 erschienen ist, passen die Hersteller ihre Produkte an den neuen Standard an. Im Folgenden werden drei Werkzeuge näher betrachtet, die im Umfeld der Daimler AG interessant sind. Die ARIS Design Platform, der Innovator for Business Analysts von MID und IBM WebSphere. Es wird betrachtet ob, und wenn ja wie weit, sie bereits die Betaversion der BPMN 2.0 unterstützen. Darüber hinaus wird die Verbindung zu anderen Diagrammen und die Umsetzbarkeit der in Kapitel 4 und 5 vorgestellten Konzepte besprochen. Ein kurzer Blick wird zudem auf die Benutzerfreundlichkeit geworfen, da diese insbesondere für Modellierer aus der Fachabteilung relevant ist. Bei allen Aspekten wird untersucht, ob sie von dem jeweiligen Werkzeug unterstützt werden (+ Symbol), teilweise unterstützt werden (o Symbol) oder nicht unterstützt werden (- Symbol).

7.1 ARIS Design Platform

Die ARIS Design Platform ist eine Produktfamilie zur Geschäftsprozessmodellierung, die von der IDS Scheer AG entwickelt wurde [IDS10b]. Die enthaltenen Programme können dazu verwendet werden Geschäftsprozesse zu modellieren, zu analysieren und zu optimieren.

Ende 2009 veröffentlichte die IDS Scheer AG das kostenlose Programm ARIS Express, welches zu der Produktfamilie der ARIS Design Platform gehört. Es war das erste Produkt der IDS Scheer AG, welches eine Unterstützung der BPMN 2.0 integriert hatte. Abbildung 7.1 zeigt die Programmoberfäche von ARIS Express beim Modellieren von BPMN Prozessdiagrammen (vgl.

Abb. 5.8). Die dort erstmals vorgestellten Funktionalitäten wurden mit dem neuesten Service Release auch in andere Produkte der ARIS Design Platform eingebaut.

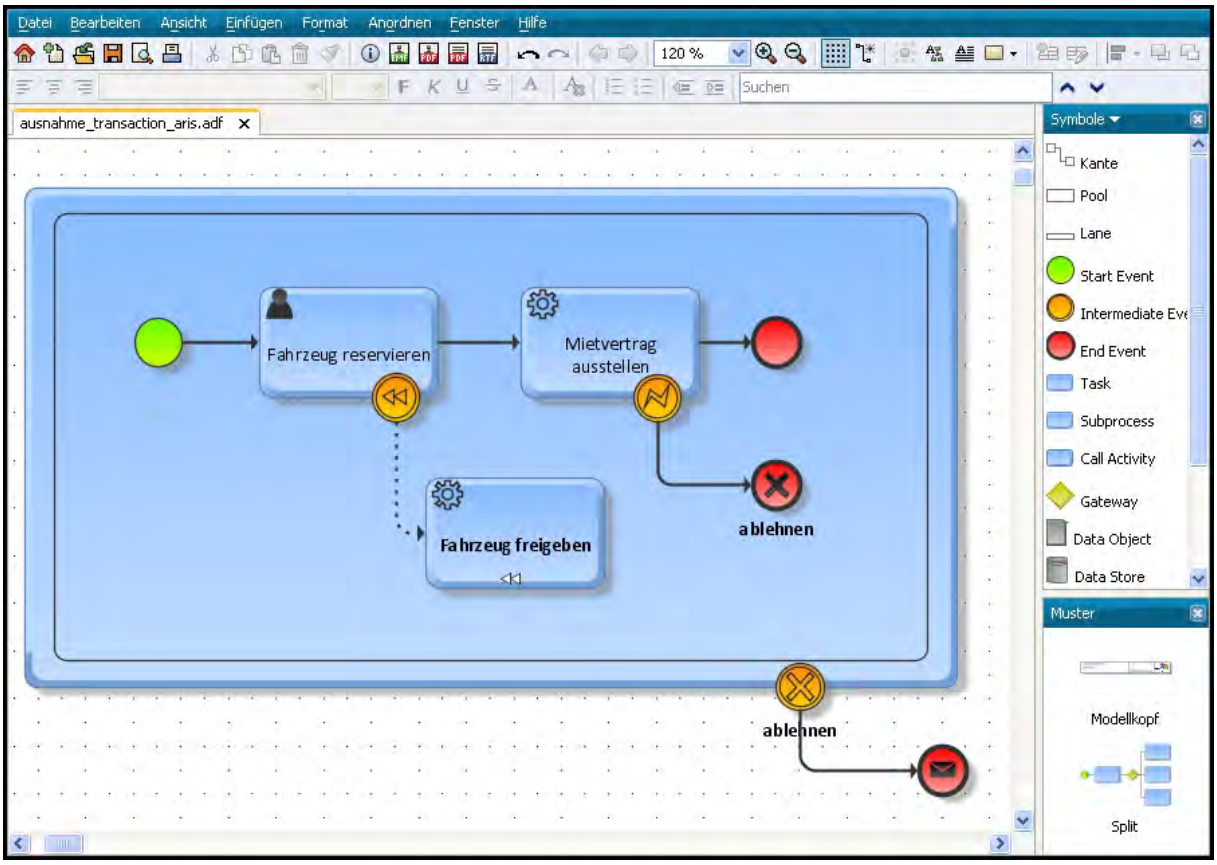


Abbildung 7.1: Neue BPMN Elemente in ARIS Express

Das kommerzielle Hauptprodukt der ARIS Design Platform ist der ARIS Business Architect. Er basiert auf der von Prof. Scheer entwickelten Architektur integrierter Informationssysteme (ARIS) [Sch92]. In der aktuellen Version 7.1 unterstützt der ARIS Business Architect seit dem Service Release 2010_5 die Betaversion der BPMN 2.0 [IDS10a]. Neben einem Diagramm zur Modellierung von BPMN Diagrammen, unterstützt er noch eine Vielzahl weiterer Diagramme. Mit diesen Diagrammen kann zum Beispiel die Organisationsstruktur eines Unternehmens modelliert werden.

Bewertung

Vollständige Umsetzung der BPMN 2.0 (o)
Die Elemente der BPMN 2.0 sind nicht vollständig umgesetzt. Der ARIS Business Architect legt bei der Modellierung von BPMN Diagrammen das Hauptaugenmerk auf die Erfassung der Geschäftsprozesse und nicht die spätere Ausführung. Aus diesem Grund sind einige Elemente

nicht oder nur rudimentär modellierbar. Dazu gehört zum Beispiel die Modellierung von Datenelementen. Diese kann nur grob vorgenommen werden (keine Input- und Output-Daten, keine Data States) [IDS10c]. In einer der nächsten Implementierungen soll die Unterstützung jedoch ausgebaut werden, sodass alle Details der BPMN 2.0 unterstützt werden.

Weitere Diagramme (+)
Neben BPMN Diagrammen können mit dem ARIS Business Architect UML Diagramme, EPKs, ER-Diagramme und eine Vielzahl weiterer Diagrammtypen erstellt werden. Im Umfeld der BPMN wird neben dem BPMN Diagramm ein weiterer Diagrammtyp angeboten, das Funktionszuordnungsdiagramm. Dieses dient dazu, Elemente aus dem BPMN Geschäftsprozessdiagramm mit Elementen in anderen Diagrammen des ARIS Business Architects zu verknüpfen. Die Verknüpfung ermöglicht es Aspekte, die in der BPMN nicht ausreichend berücksichtigt werden, zu modellieren. Abbildung 7.2 zeigt ein Funktionszuordnungsdiagramm, mit dem der `Activity` "Kunde überprüfen" eine in einem anderen Diagramm modellierte Benutzermaske zugeordnet wird. Zudem existiert die Möglichkeit, einzelne Elemente direkt im BPMN Diagramm durch sogenannte "Assignments" mit anderen Diagrammen zu verknüpfen.

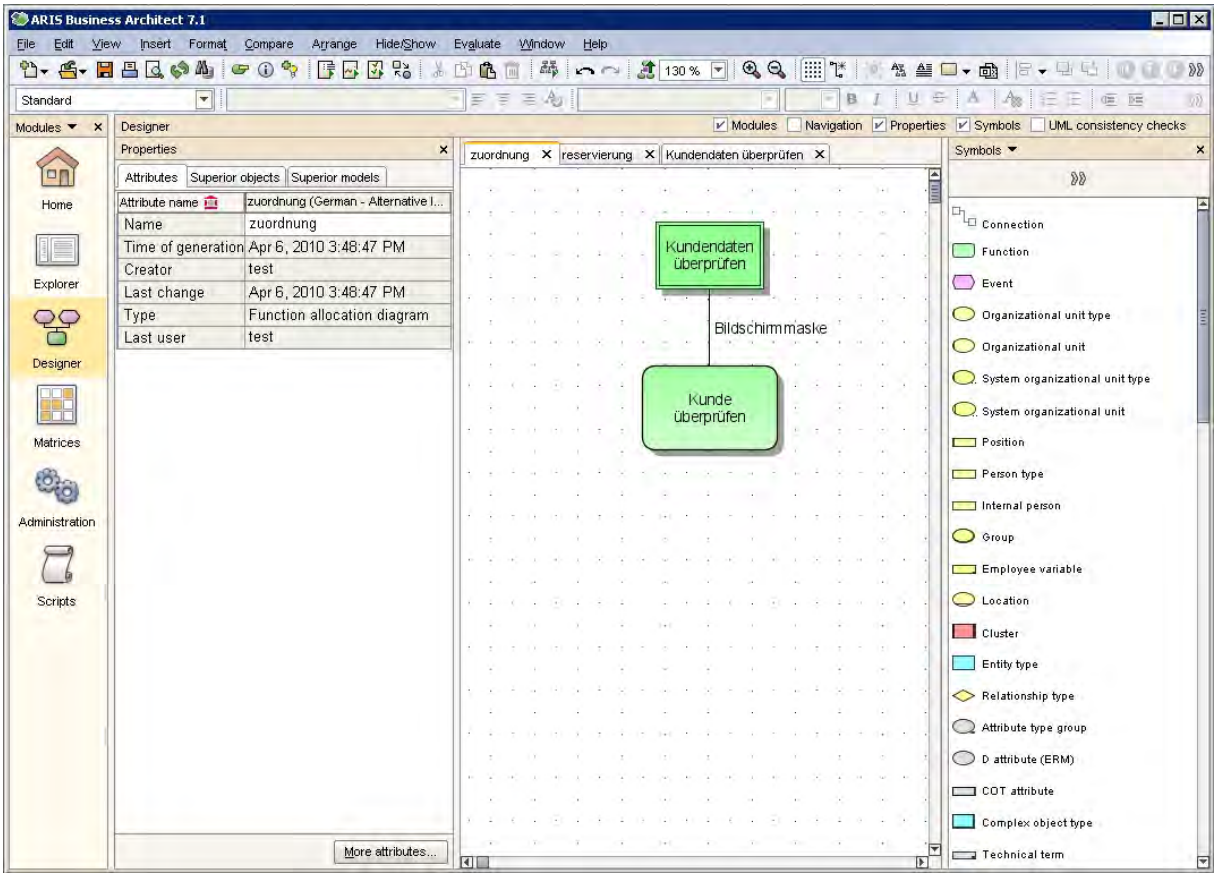


Abbildung 7.2: Verknüpfung von BPMN Elemente mit anderen Elementen in ARIS

Daten (o)
Daten können aus fachlicher Sicht gut modelliert werden. Die Daten werden durch `Data Objects` innerhalb des mit BPMN modellierten Geschäftsprozesses modelliert. In externen Diagrammen, die mit den `Data Objects` verknüpft werden, können die Daten weiter detailliert werden. Auf Systemebene kann die Verarbeitung der Daten jedoch nicht präzise genug beschrieben werden. Eine Möglichkeit Ein- und Ausgabedaten von Prozessschritten formal zu beschreiben ist nicht vorhanden. Gleiches gilt für den Datenbezug von Verzweigungsstellen.

Bearbeiterzuordnung (o)
Um eine Organisationsstruktur zu modellieren, bietet der ARIS Business Architect ein eigenes Diagramm an, das Organigramm. Die dort definierten Strukturen können über ein Funktionszuordnungsdiagramm den `Activities` im BPMN Diagramm zugeordnet werden. Die Zuordnung kann so wie in Abschnitt 4.1.1 gezeigt grafisch vorgenommen werden. Es können jedoch nicht alle, sondern nur einfache Regel, wie zum Beispiel die Zugehörigkeit zu einer Organisationseinheit, modelliert werden. Zudem kann durch das Fehlen von logischen Operatoren keine Verknüpfung der Regeln gebildet werden, was die Zuordnung zusätzlich einschränkt.

Ausnahmebehandung (+)
Die in der BPMN vorhandenen Elemente zur Ausnahmebehandlung können modelliert werden. Um im fachlichen Modell eine strukturierte textuelle Dokumentation der Ausnahmen vorzunehmen, können den Elementen neue Attribute zugewiesen werden.

Flexibilitätsstellen (o)
Im ARIS Business Architect ist es nicht möglich, eigene Elemente zu definieren. Deshalb können auch keine neuen Elemente für die Markierung von Flexibilitätsstellen definiert werden. Es gibt jedoch die Möglichkeit, Flexibilitätsstellen durch Freitext sowie Rechtecke und Ellipsen, die beliebig im Diagramm platziert werden können, zu markieren. Eine Semantik ist bei diesen Konstrukten jedoch nicht hinterlegt. Um Geschäftsregeln aus den Geschäftsprozessen zu ziehen, wird der ARIS Business Rules Designer angeboten.

Benutzerfreundlichkeit (o)
Um ein BPMN Diagramm zu modellieren, muss der Modellierer das gewünschte Element aus der rechten Symbolleiste auswählen. Anschließend kann er es durch einen Klick im Diagramm platzieren. Die Symbolleiste ist anpassbar, sodass die angezeigten Elemente je nach Fähigkeiten der Modellierer und die Sichtweise (Fach- oder Systemmodell) geändert werden können. Die Arbeit mit dem ARIS Business Architect ist nicht immer intuitiv, da das übliche Windows "Look and Feel" nicht immer Anwendung findet. Erleichtert wird die Modellierung durch einige Funktionen [SH09] wie etwa eine automatische Größenanpassung von `Pools` und `Lanes`. Zudem werden Einschränkungen in der BPMN automatisch erkannt und bei fehlerhafter Modellierung korrigierend eingegriffen. Es ist zum Beispiel nicht erlaubt zwei `Activities`, die sich in unterschiedlichen `Pools` befinden, durch einen Sequenzfluss zu verbinden. Wird ein `Event` an einen Nachrichtenfluss angebunden, wird er automatisch zu einem `Message Event`. Das modellierte Diagramm wird durch eine Syntaxvalidierung geprüft, um den Modellierer auf syntaktische Fehler hinzuweisen. Es wird zum Beispiel angezeigt, wenn ein `Message Event` keinen eingehenden Nachrichtenfluss besitzt. Im Lieferumfang sind insgesamt 70 Syntaxregeln bereits vorhanden. Die Syntaxvalidierung basiert auf JavaScripts, die auf einer internen Scripting-Engine laufen. So können bei Bedarf Regeln abgeändert, gelöscht oder hinzugefügt werden.

Fazit

Die ARIS Design Platform unterstützt derzeit noch nicht die vollständige Betaversion der BPMN 2.0. Diese Schwachstelle wird nach der Veröffentlichung der endgültigen BPMN 2.0 wohl behoben werden. Es ist geplant, die neue Spezifikation zeitnah in den Produkten der ARIS Design Platform umzusetzen [SH09]. Einem Modellierer aus der Fachabteilung dürfte es jedoch nicht leicht fallen, bei den vielen Funktionen den Überblick zu bewahren. Herauszuheben sind die vielfältigen Diagrammtypen und Verknüpfungsmöglichkeiten zu diesen.

7.2 MID Innovator for Business Analysts

Der Innovator for Business Analysts wurde von der MID GmbH entwickelt [MID10a]. Er richtet sich an Business Analysten, die mit dessen Hilfe die Prozesse ihrer Kunden modellieren können. Als Business Analyst wird dabei ein Mitarbeiter bezeichnet, der als Mittler zwischen IT-Abteilung und Fachbereich Anforderungen der Fachbereiche aufnimmt und diese mit der IT-Abteilung abstimmt.

Der Innovator ist lediglich ein Modellierungswerkzeug und bietet keine Process-Engine. Demzufolge können mit dem Innovator erstellte Diagramme nicht direkt mit diesem ausgeführt werden, sondern müssen exportiert und anschließend in einer Process-Engine der Wahl importiert werden.

Bewertung

Vollständige Umsetzung der BPMN 2.0 (+)
Die in der Betaversion der BPMN 2.0 Spezifikation definierten Elemente werden inklusive ihrer Attribute alle vom Innovator unterstützt. Zusätzlich bietet er eine Erweiterungsmöglichkeit an, mit der die Elemente der BPMN um weitere Attribute erweitert werden können. Als Exportformate unterstützt der Innovator neben BPEL auch das neue BPMN-XML-Format. Zudem werden alle Exportformate über Templates erzeugt, die vom Benutzer selbst angepasst werden können.

Weitere Diagramme (+)
Mit dem Innovator können über BPMN Diagramme hinaus UML Klassendiagramme, UML Anwendungsfalldiagramme, Maskenflussdiagramme und Whiteboard Diagramme erstellt werden. Diese können mit den in BPMN modellierten Geschäftsprozessen beziehungsweise mit einzelnen Teilen aus diesen verknüpft werden.
Mit Hilfe des Maskenflussdiagramms lassen sich Oberflächen erstellen. Dabei besteht die Möglichkeit, die einzelnen Eingabefelder der Oberfläche mit den spezifizierten Datenobjekten aus dem BPMN Diagramm zu verknüpfen.
Um Querverbindungen zwischen Diagrammen und einzelnen Elementen daraus herzustellen, bietet der Innovator Whiteboard Diagramme an. Ein Whiteboard Diagramm ist ein Diagramm, in das andere Diagramme eingebettet werden können. Durch Linien und Pfeile zwischen den einzelnen Elementen können Querverbindungen aufgezeigt werden. Die eingebetteten Diagramme werden dabei immer automatisch auf dem aktuellen Stand gehalten. Die eingezeichneten Querverbindungen sind nur visuelle Elemente und mit keiner Semantik belegt. Sie dienen lediglich dazu,

Zusammenhänge zu verdeutlichen und die Verständlichkeit der Modelle zu erhöhen. Abbildung 7.3 zeigt ein Whiteboard Diagramm, bei dem das Datenelement "TNdaten" aus einem BPMN Diagramm mit der Klasse "Teilnehmerdaten" aus einem UML Klassendiagramm verknüpft ist.

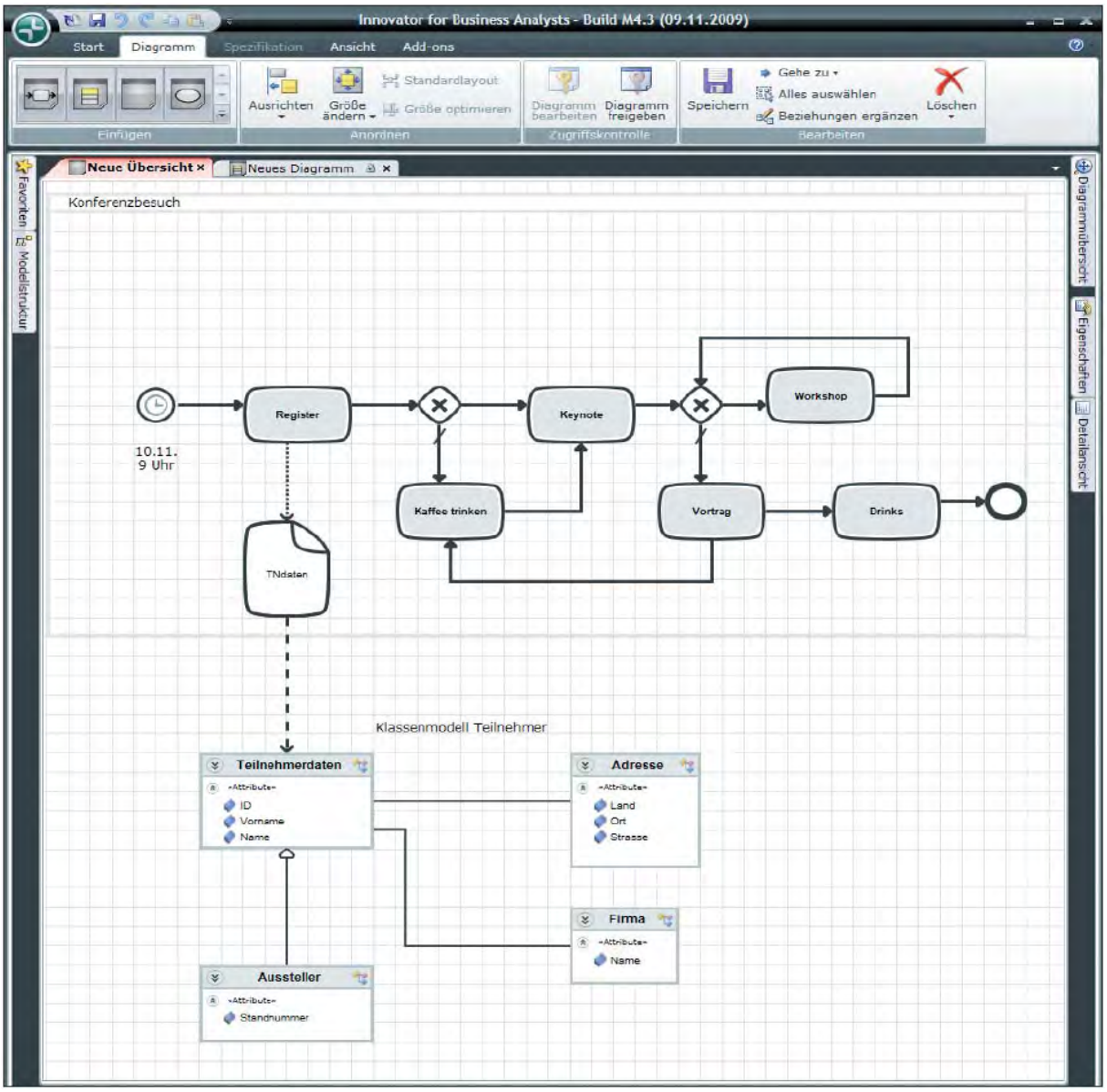


Abbildung 7.3: Whiteboard Diagramm im MID Business Modeller

Daten (+)
Die Detaillierung von in BPMN Diagrammen modellierten Datenelementen kann durch UML Klassendiagramme vorgenommen werden. Die Datenelemente aus dem Prozessdiagramm werden dabei mit Klassen aus dem UML Klassendiagramm verbunden.

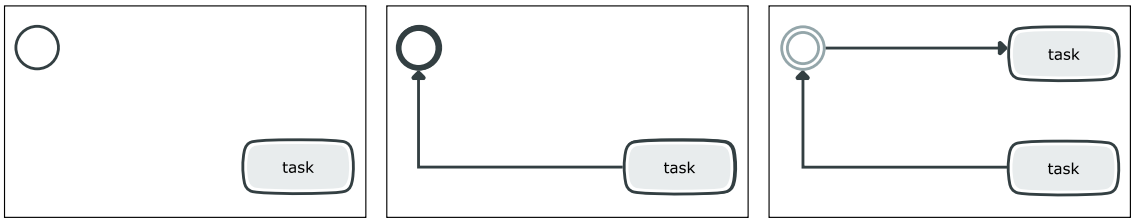
Bearbeiterzuordnung (o)

Der Innovator bietet keinen eigenen Diagrammtyp an, um die Organisationsstruktur eines Unternehmens zu beschreiben. Es besteht jedoch die Möglichkeit, diese mit Hilfe eines UML Klassendiagramms zu modellieren [Sch04]. Dieses kann dazu um Stereotypen erweitert werden.
Die Zuordnung der Bearbeiter zu den einzelnen Prozessschritten kann nicht durch Regeln, wie in Abschnitt 4.1.2 beschrieben, durchgeführt werden. Es besteht lediglich die in der BPMN integrierte Möglichkeit der Zuordnung über `Swimlanes` und `Resources`.

Ausnahmebehandung (+)
Die für die Ausnahmebehandlung nötigen Elemente der BPMN 2.0 sind vollständig umgesetzt. Eine Ausnahmebehandlung kann so gut durchgeführt werden.

Flexibilitätsstellen (o)
Um Flexibilitätsstellen markieren zu können, existieren keine eigenen Elemente. Es ist jedoch möglich, eigene Elemente zu definieren und mit diesen Flexibilitätsstellen zu markieren.
Um Geschäftsregeln beschreiben zu können, ist die Verbindung zu einer Business-Rule-Engine wünschenswert. Diese ist aktuell noch nicht vorhanden, soll jedoch in einer späteren Version realisiert werden.

Benutzerfreundlichkeit (+)
Um dem Modellierer eine leichte Benutzbarkeit zu bieten, ist die Software mit einer Ribbon-Bar ausgestattet (siehe Abb. 7.3). Dieses Bedienkonzept, das Menüsteuerung und Symbolleiste miteinander verbindet, ist aus den aktuellen Office-Applikationen ab Version 2007 bekannt. Um Elemente dem Diagramm hinzuzufügen, können diese einfach in das Diagramm gezogen werden oder vom Vorgängerelement durch einen einfachen Klick hinzugefügt werden.
Zusätzlich unterstützt der Innovator den Modellierer, indem er beispielsweise den Typ eines `Events` aus dem Kontext heraus selbst bestimmen kann (siehe Abb. 7.4). Ein unverbundenes `Event` wird als `Start-Event` modelliert (siehe Abb. 7.4a). Wird dieses mit einem eingehenden Sequenzfluss verknüpft, wird daraus automatisch ein `End-Event` (siehe Abb. 7.4b). Wird aus dem `End-Event` ein ausgehender Sequenzfluss modelliert, wird daraus automatisch ein `Intermediate-Event` (siehe Abb. 7.4c).



(a) Start-Event (b) End-Event (c) Intermediate-Event

Abbildung 7.4: Automatische Anpassung des Eventtyps

Um zu garantieren, dass die erstellten Modelle auch korrekt sind, bietet der Innovator eine Syntaxvalidierung an. Diese besteht aus über 10.000 Prüfroutinen, die die Diagramme auf ihre syntaktische Korrektheit prüfen. Es wird zum Beispiel erkannt, wenn in einem Diagramm ein `Start-Event` modelliert ist, ein `End-Event` jedoch fehlt, was laut Spezifikation nicht erlaubt ist.

Fazit

Insgesamt bietet der Innovator for Business Analysts von MID eine intuitive Oberfläche an. Diese ermöglicht es auch unerfahrenen Modellierern, ohne eine lange Einarbeitungszeit, Geschäftsprozessmodelle zu erstellen. Durch die Verbindung mit anderen Diagrammen können Informationen, die nicht in einem BPMN Diagramm modelliert werden können, hinterlegt werden. Insbesondere bei der Bearbeiterzuordnung zeigt der Innovator for Business Analysts jedoch Schwächen, da lediglich die in der Spezifikation der BPMN enthaltenen Konzepte umgesetzt wurden. Diese sind für eine gute Bearbeiterzuordnung jedoch nicht ausreichend.

7.3 IBM WebSphere

IBM stellt mit dem Produktpaket WebSphere Software für das Geschäftsprozessmanagement zur Verfügung [IBM10]. Die einzelnen Produkte decken dabei von der Modellierung der Geschäftsprozesse, über deren Analyse, die Ausführung und die Überwachung alles ab, was für das Geschäftsprozessmanagement benötigt wird [Pei09]. Drei Produkte bilden die Grundlage, der Business Modeller, der Process Server und der Business Monitor. Der Business Modeller wird dazu verwendet, Geschäftsprozesse zu modellieren und zu analysieren. Er richtet sich an die Fachabteilungen. Abbildung 7.5 zeigt den mit dem Business Modeller modellierten Prozess der Abrechnung aus dem Anwendungsbeispiel (vgl. Abb. 1.4). Mit Hilfe des WebSphere Integration Developers, einem weiteren Produkt aus dem WebSphere Produktpaket, lassen sich ebenfalls Prozessmodelle erstellen, jedoch auf einer weitaus technischeren Ebene. Er wird vorwiegend von den IT-Abteilungen verwendet. Die modellierten Geschäftsprozesse können anschließend mit dem Process Server ausgeführt werden. Die Ausführung kann mit dem Business Monitor überwacht und ausgewertet werden. Über diese drei Produkte hinaus, existieren additive Produkte, mit denen der Funktionsumfang weiter erhöht werden kann. Darüber hinaus lässt sich IBM WebSphere auch mit anderen Produkten aus dem IBM Software Angebot kombinieren. Im Zuge der Softwareentwicklung ist zum Beispiel eine Kombination mit IBM Rational Rose interessant [IBM09].

Bewertung

Vollständige Umsetzung der BPMN 2.0 (o)
Der WebSphere Process Server basiert auf BPEL. Da BPEL keine grafische Notation besitzt, wird für die grafische Modellierung eine an die BPMN angelehnte Notation verwendet. Diese wird automatisch in BPEL übersetzt. Da die BPMN nicht direkt in BPEL überführt werden kann, wird nicht die BPMN, sondern nur eine abgeänderte Version dieser benutzt [ODHA06, VVLM08]. Trotzdem bietet der Business Modeller die Möglichkeit, BPMN Diagramme zu erstellen (siehe Abb. 7.6).

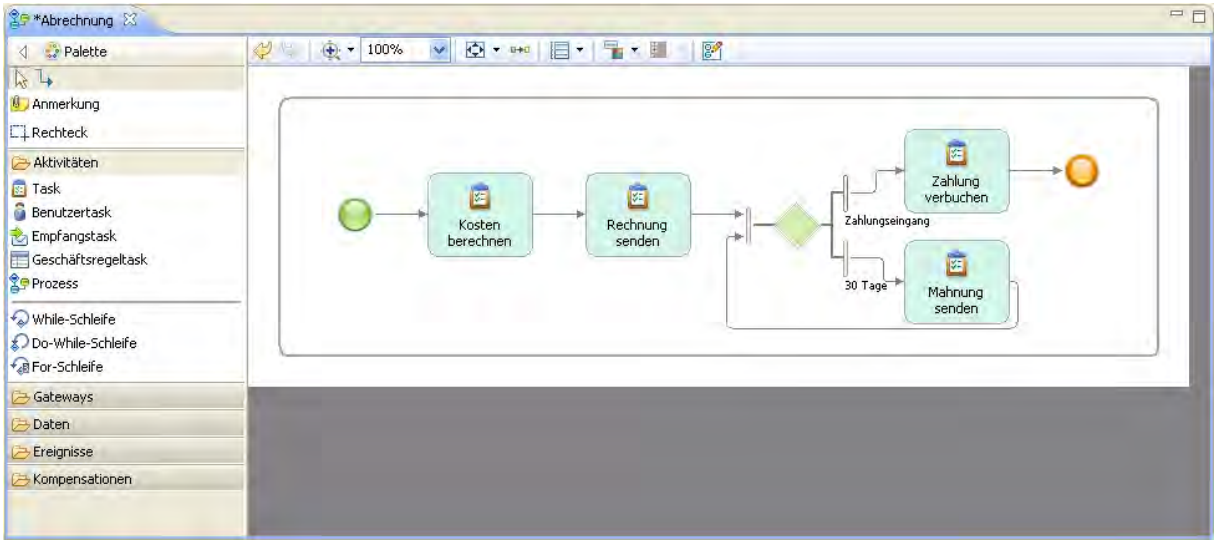


Abbildung 7.5: Prozessmodlelierung im Business Modeller

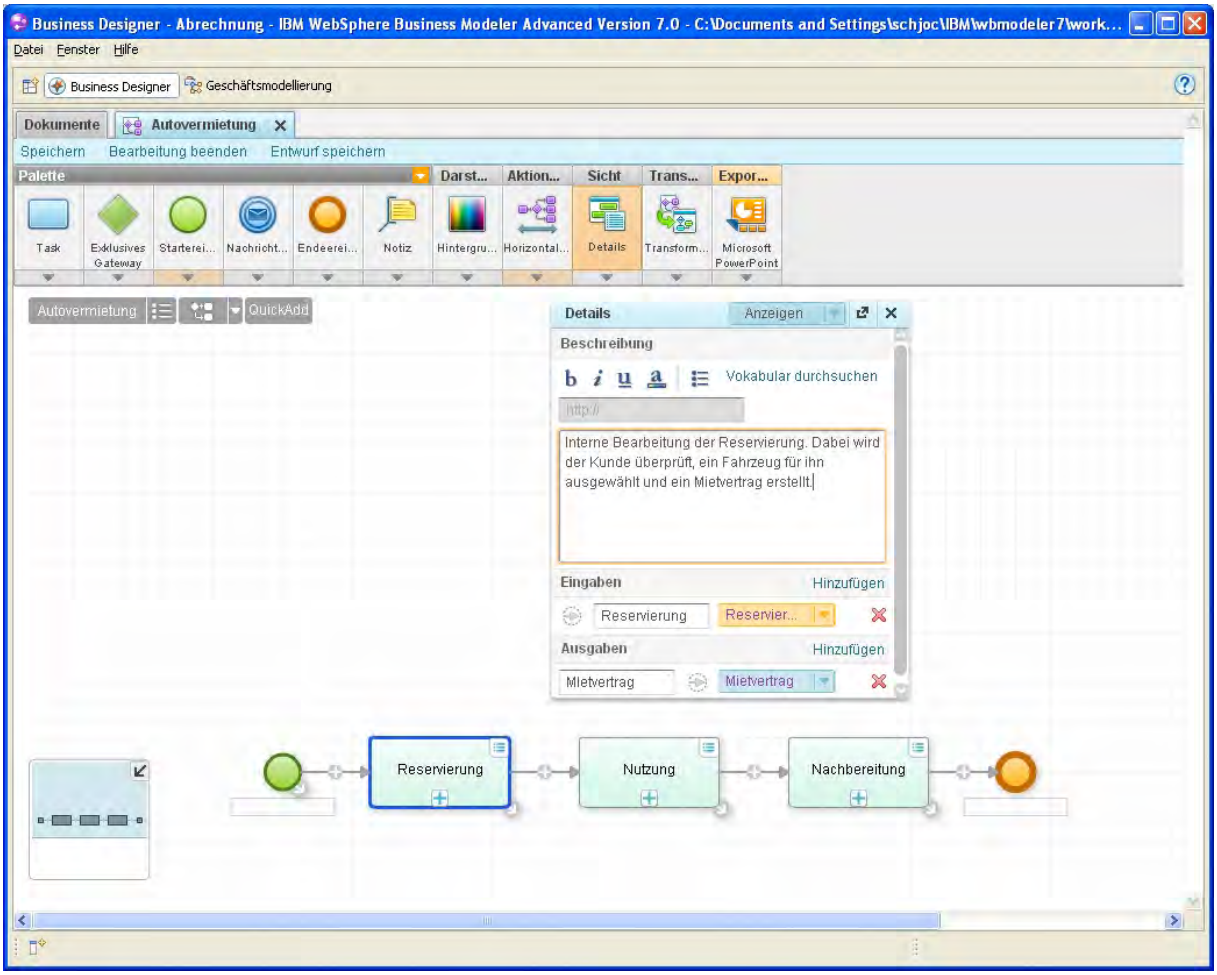


Abbildung 7.6: BPMN Modellierung im WebSphere Business Modeller

In der neuen Version 7.0 können bereits BPMN 2.0 Diagramme erstellt werden Die Betaversion der BPMN 2.0 wurde bisher jedoch noch nicht vollständig, sondern nur auf dem sogenannten "Level 1" [Sil09] umgesetzt. Das bedeutet, dass lediglich einige wenige einfache Elemente unterstützt werden. Diese ermöglichen es der Fachabteilung, ihre Geschäftsprozesse zu modellieren. Der Detaillierungsgrad reicht jedoch nicht aus, um ein ausführbares Prozessmodell zu erstellen. Die so erstellten BPMN Prozessmodelle können automatisch in die BPMN ähnliche Darstellung transferiert werden, die später wiederum in BPEL Code überführt wird. Dort kann der modellierte Geschäftsprozess weiter verfeinert werden und, nachdem er in BPEL überführt wurde, durch den Process Server zur Ausführung gebracht werden. Zudem besteht die Möglichkeit die erstellten Diagramme im neuen BPMN 2.0 XML-Austauschformat zu exportieren.

Weitere Diagramme (o)
Über das Prozessdiagramm hinaus, können auch weitere Informationen, wie zum Beispiel Daten beschrieben werden. Diese Beschreibung erfolgt jedoch meist nicht in Diagrammen, sondern durch Masken. Eine solche Beschreibung ist mindestens genauso detailliert, es fehlt jedoch eine übersichtliche Darstellung der Zusammenhänge, zwischen den einzelnen Elementen. An den Stellen wo Diagramme eingesetzt werden, sind dies proprietäre Lösungen.

Daten (+)
Daten können als sogenannte Geschäftsobjekte definiert und mit den Datenelementen innerhalb des Prozessdiagramms verknüpft werden. Für die Detaillierung der Geschäftselemente können Attribute, Regeln, die das Verhalten steuern, und verschiedene Status, die das Objekt annehmen kann, definiert werden. Mittels Zuordnungen können Geschäftsobjekte in andere Geschäftsobjekte übersetzt werden. Bei den einzelnen Prozessschritten ist eine genaue Definition der Eingabe- und Ausgabedaten möglich. Zudem kann eine Eingabe- und eine Ausgabelogik definiert werden. Erst wenn eine der definierten Bedingungen erfüllt ist, kann der Prozessschritt starten beziehungsweise die Ausgabe generiert werden.

Bearbeiterzuordnung (+)
Insbesondere was die Zuordnung von Bearbeitern zu Prozessschritten betrifft, bietet IBM WebSphere gute Produkte an. Mit dem WebSphere Member Manager und People Finder kann eine Organisationsstruktur modelliert werden. Durch sogenannte "Staff Verbs" kann die Zuordnung von Bearbeitern zu einzelnen Prozessschritten vorgenommen werden. Diese sind sehr ähnlich zu dem in Abschnitt 4.1.1 vorgestellten Ansatz. Die Anzahl an Staff Verbs ist standardmäßig relativ begrenzt, lässt sich aber durch benutzerdefinierte Staff Verbs erweitern. Abbildung 7.7 zeigt die Zuordnung mittels Staff Verbs im Integration Developer. Im Auswahlfeld "Staff Group (Verb)" kann das gewünschte Staff Verb ausgewählt werden. In dem gezeigten Fall soll der Bearbeiter der Vorgesetzte eines Mitarbeiter sein, der anhand einer ID ermittelt wird. Nach der Auswahl des gewünschten Staff Verbs, wird ein Erklärungstext angezeigt und es können die benötigten Parameter gesetzt werden. Im gezeigten Fall muss lediglich die ID des Mitarbeiters angegeben werden. Diese wird aus dem vorhergehenden Prozessschritt "InputJourneyData" ermittelt.

Ausnahmebehandung (o)
Die bisher in der BPMN umgesetzten Elemente erlauben keine Ausnahmebehandlung. Bei der Modellierung in der von WebSphere benutzten Notation kann eine Ausnahmebehandlung durchgeführt werden. Dazu existieren die Elemente "Überwachungsfunktion" und "Zeitgeber". Zusätzlich können für bereits durchgeführte Prozessschritte Kompensationsaktivitäten angegeben wer-

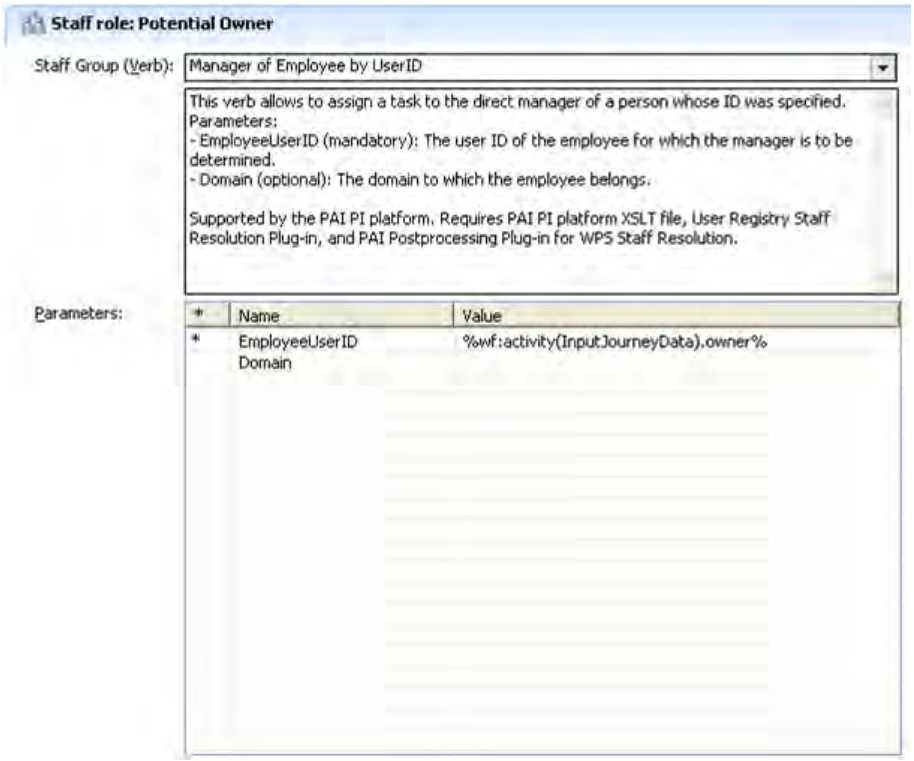


Abbildung 7.7: Bearbeiterzuordnung in IBM WebSphere durch Staff Verbs

den, die in einem Ausnahmefall ausgeführt werden, um die durchgeführten Änderungen zurückzusetzen.

Flexibilitätsstellen (o)
IBM WebSphere bietet keine Möglichkeit eigene Elemente zu definieren. Eine Markierung von Flexibilitätsstellen kann jedoch durch angeheftete Notizen vorgenommen werden.
Für die Umsetzung von Flexibilität bietet WebSphere mehrere Produkte an. So kann etwa mit dem WebSphere ESB der Aufruf eines Services entkoppelt werden. Mit dem Produkt ILOG jRules wird ein Business Rule Management System angeboten, mit dem Geschäftsregeln verwaltet werden können. Die dort definierten Regeln können direkt vom modellierten Geschäftsprozess aufgerufen werden.

Benutzerfreundlichkeit (o)
Web Sphere baut auf der Eclipse Umgebung auf. Modellierern, die bereits mit dieser Oberfläche vertraut sind, fällt die Einarbeitung leicht. Die Verwendung von Eclipse als Grundlage zeigt bereits die technische Ausrichtung von WebSphere, mit dem Ziel, die modellierten Prozesse auszuführen. Diese technische Ausrichtung erschwert die Arbeit für Modellierer aus der Fachabteilung.
Die Prozessdiagramme können durch Drag and Drop bearbeitet werden. Das Layout des Diagramms muss nicht manuell angepasst werden, sondern wird automatisch erstellt.

Fazit

Insgesamt bietet IBM mit WebSphere ein umfassendes Produktpaket zum Geschäftsprozessmanagement. Durch die Vielzahl an additiven Produkten, können viele der in dieser Arbeit beschriebenen Ansätze umgesetzt werden. Die Unterstützung der BPMN 2.0 ist jedoch noch nicht in allen Aspekten befriedigend. Da IBM jedoch Mitglied im Standardisierungsgremium der BPMN 2.0 ist, kann davon ausgegangen werden, dass nach der endgültigen Standardisierung der BPMN 2.0 diese in WebSphere weitestgehend umgesetzt wird. Aufgrund der technischen Ausrichtung eignet sich WebSphere weniger für die fachliche Modellierung. Durch die integrierte Process-Engine ist der Übergang zu einem ausführbaren Modell einfach durchzuführen.

7.4 Zusammenfassung

Tabelle 7.1 zeigt eine Übersicht der Bewertung der einzelnen Aspekte der drei evaluierten Werkzeuge.

	IDS Scheer	MID	IBM
Vollständigkeit der BPMN 2.0	o	+	o
Verknüpfung zu weiteren Diagrammen	+	+	o
Daten	o	+	+
Bearbeiterzuordnung	o	o	+
Ausnahmebehandlung	+	+	o
Flexibilität	o	o	o
Benutzerfreundlichkeit	o	+	o

Tabelle 7.1: Vergleich der untersuchten Werkzeuge

Da alle untersuchten Aspekte für die Prozessmodellierung wichtig sind, werden sie von allen Werkzeugen zumindest teilweise unterstützt. Für die Modellierung von Geschäftsprozessen mit der BPMN 2.0 eignet sich derzeit der Innovator for Business Analysts von MID am besten. Er unterstützt alle Elemente der BPMN 2.0 und bietet eine einfache Oberfläche, die es auch IT unerfahrenen Modellierern aus den Fachabteilungen ermöglicht sich schnell einzuarbeiten.

Da MID jedoch keine Process-Engine anbietet, muss für die Ausführung des Geschäftsprozesses das Modell exportiert und in einer Process-Engine eines anderen Herstellers importiert werden. Die Übernahme des Modells wird durch das standardisierte Austauschformat der BPMN 2.0 vereinfacht. Trotzdem bedeutet ein Werkzeugbruch potentielle Probleme. Soll der modellierte Geschäftsprozess mit Hilfe einer Process-Engine ausgeführt werden, ist es deshalb überlegenswert, den Geschäftsprozess mit IBM WebSphere zu modellieren. Durch die integrierte Process-Engine wird so ein Werkzeugbruch vermieden.

8 Diskussion

In diesem Kapitel werden verwandte Arbeiten diskutiert und in Zusammenhang mit dieser Arbeit gebracht. Neben MDA (Model-Driven Architecture) basierten Ansätzen werden Modellierungsmethoden von Herstellern diskutiert. Anschließend werden Modellierungssprachen für Geschäftsprozesse betrachtet.

Zahlreiche auf MDA basierende Ansätze beschreiben Vorgehensweisen, um aus fachlichen Geschäftsprozessen ausführbare Modelle zu generieren. [All07] beschreibt eine Modelltransformation mittels Mustern (Pattern), die fachliche Objekte detaillieren und die Überführung in das Zielmodell beschreiben. Um die Transformationen durchführen zu können, werden unter Verwendung der Patterns Tranformationsregeln definiert.

[BFF+08] beschreibt ebenfalls Patterns, um eine Transformation durchzuführen. Zunächst werden Patterns auf fachliche Prozessmodelle (BPMN) angewandt. Das resultierende Modell (erweitertes BPMN) wird durch eine automatische Transformation in ein ausführbares Grundmodell überführt (Pseudo-BPEL). Dieses wird mit Hilfe weiterer technischer Patterns in ein ausführbares Modell (BPEL) überführt.

Bei auf MDA basierenden Ansätze werden Patterns auf fachliche Modelle angewandt. Wird jedoch eine freie Modellierbarkeit des Systemmodells gefordert, sind solche Ansätze nicht realisierbar. Geschäftsprozesse, die auf fachlicher Ebene lediglich vage beschrieben wurden, müssen auf Systemebene strukturell angepasst werden. Dies mittels automatisch anwendbarer Patterns zu realisieren ist theoretisch möglich, praktisch jedoch zu aufwändig. Transformationen wie etwa das Einfügen von zusätzlichen Aktivitäten sind schwer zu realisieren, da im Fachmodell keine entsprechenden Aktivitäten definiert sind. Eine Wiederverwendung der definierten Patterns ist zudem nicht möglich, da Geschäftsprozesse individuell sind. Für jeden Geschäftsprozess müssen deshalb individuelle Patterns definiert werden.

Bestehende Ansätze beschäftigen sich lediglich mit einer automatischen Transformation zwischen den Modellen. Eine Diskussion über die Granularität der Modellierung auf den verschiedenen

Ebenen wird nicht geführt. So wird nicht beschrieben, was einem Fachmodellierer zugetraut werden kann zu modellieren und wo seine Grenzen liegen (Frontloading). Ebenso wenig wird betrachtet, wie die Modellierung einzelner Anforderungen tatsächlich realisiert werden kann. In dieser Arbeit wird sowohl der Aspekt des Frontloadings sowie die Umsetzung einzelner Anforderungen (Bearbeiter, Daten, Ausnahmen, Flexibilität) betrachtet.

Um eine freie Modellierung des Systemmodells zu ermöglichen, führen Hersteller von Modellierungswerkzeugen eigene Vorgehensweisen ein. Dazu wird zwischen fachlicher Ebene und dem ausführbaren Modell eine neue Ebene eingeführt. Diese ist vergleichbar mit der in dieser Arbeit verwendeten Systemebene.

Die Firma MID etwa stellt mit M3 eine eigene Modellierungsmethodik bereit [DPRW08]. Sie besteht ebenfalls aus drei Ebenen, die in etwa mit den drei in dieser Arbeit verwendeten Ebenen gleichzustellen sind. M3 basiert auf der MDA und ist im Innovator umgesetzt. Die Modellierung beruht auf UML Diagrammen, die durch Stereotypen erweitert wurden. Von anderen Herstellern wie etwa IBM [AGA+08, Ars04] oder IDS Scheer [IDS05, AB07] werden ebenfalls Vorgehensmethoden beschrieben, um fachliche Modelle in ausführbare Applikationen zu überführen. Dazu wird ebenfalls eine Zwischenebene eingeführt.

Die Transformation zwischen den Modellebenen ist bei den beschriebenen Vorgehensmethoden jedoch lediglich in eine Richtung definiert. Komplexe Abhängigkeiten der Diagrammelemente zwischen den unterschiedlichen Modellebenen sind nicht nachvollziehbar. Dadurch kommt es bei Änderungen in einer Modellebene zu Inkonsistenzen. Durch die Verwendung einer einheitlichen Notation und einem Abbildungsmodell zwischen den Ebenen können diese beseitigt werden [BBR10a, BBR10b].

Zur Modellierung von Geschäftsprozessen existieren verschiedene Modellierungssprachen wie etwa BPMN 2.0 [OMG09b], EPK [KNS92], UML Aktivitätsdiagramme [OMG09c], Petri-Netze [Pet62] oder YAWL [AH05]. Um Geschäftsprozesse ausführen zu können, wird eine Ausführungssprache wie etwa BPEL [OAS07] benötigt. Aus diesem Grund ist eine Transformation von der fachlichen Modellebene (bspw. EPK) in eine Ausführungssprache (bspw. BPEL) nötig. Neben einer direkten Transformation ist auch eine Transformation über eine zusätzliche Modellebene (bspw. eEPK) denkbar. Für die Transformation existieren viele Ansätze. [ZD08] beschreibt die Transformation von UML Aktivitätsdiagrammen nach BPEL, [SI07] die Transformation von EPKs nach BPEL. Transformationen von BPMN nach BPEL werden bereits im BPMN-Standard [OMG09a, OMG09b] als auch in weiteren Arbeiten [Whi05, ODHA06] beschrieben. Um eine Transformation zwischen den verschiedenen Sprachen durchführen zu können, muss das Quellmodell jedoch durch Restriktionen (bspw. Zyklenfreiheit) eingeschränkt werden.

Im Unterschied zur Version 1 verfügt die BPMN 2.0 über eine eigene Ausführungssemantik, sodass mit der BPMN modellierte Geschäftsprozesse durch eine Process-Engine direkt ausgeführt werden können. Eine Überführung in eine Ausführungssprache wie etwa BPEL ist dazu nicht erforderlich. Gleichzeitig eignet sie sich, wie auch die EPK, gut für eine fachliche Modellierung. So eignet sich die BPMN 2.0 im Gegensatz zu den anderen Modellierungssprachen für eine durchgängige Modellierung der Geschäftsprozesse von der fachlichen Ebene bis hin zur IT-Applikation [BBR10a]. Mit der BPMN können jedoch nicht alle Anforderungen an die Geschäftsprozessmodellierung (vgl. Abschnitt 1.3) zufriedenstellend abgedeckt werden. Deshalb werden in dieser

Arbeit verschiedene Erweiterungen beschrieben, um zum Beispiel die Bearbeiterzuordnung und die Markierung von Flexibilitätsstellen zufriedenstellend vornehmen zu können.

9 Zusammenfassung

Im Rahmen dieser Arbeit wird untersucht, in wie weit sich die BPMN 2.0 dazu eignet, fachliche Anforderungen an die Geschäftsprozessmodellierung zu dokumentieren und bis hin zu einem ausführbaren Modell weiter zu verfeinern. Wo die BPMN 2.0 Schwachstellen zeigt, werden Erweiterungen vorgeschlagen. Der Vorteil einer durchgängigen Modellierung mit der BPMN 2.0 liegt in einer einfacheren Transformation der Modelle zwischen der Fach-, System- und ausführbaren Ebene. Zudem wird durch die Verwendung einer einheitlichen Modellierungssprache auf allen Ebenen das Business-IT-Alignment verbessert.

Dazu wird zunächst ein vereinfachtes Anwendungsszenario beschrieben (siehe Abschnitt 1.2). Ausgehend davon werden in Abschnitt 1.3 Anforderungen an die durchgängige Geschäftsprozessmodellierung beschrieben.

In Kapitel 2 werden allgemeine Vorgehensweisen zur Modellierung von Geschäftsprozessen vorgestellt. Darüber hinaus werden Modellierungssprachen vorgestellt, die zur Modellierung von Geschäftsprozessen verwendet werden können. Die in dieser Arbeit speziell betrachtete BPMN 2.0 wird in Kapitel 3 ausführlich beschrieben.

Den Hauptteil dieser Arbeit bildet die Entwicklung und Bewertung von Ansätzen zu einer durchgängigen Modellierung von fachlichen Anforderungen und deren Abbildung in ein sogenanntes Systemmodell [BBR10a, BBR10b]. Auf Fach- und Systemebene werden jeweils potentielle Ansätze zur Modellierung von Bearbeiterzuordnung, Daten, Ausnahmen und Flexibilitätsstellen im Prozessmodell entwickelt. Diese werden evaluiert und anschließend ihre Realisierbarkeit in BPMN 2.0 geprüft. Ferner werden Werkzeuge zur Modellierung von Geschäftsprozessen mit der BPMN 2.0 evaluiert.

In Kapitel 4 wird die fachliche Modellierung der Anforderungen beschrieben. Ein Schwerpunkt wird dabei auf das Frontloading gelegt. Für jede der Anforderungen werden verschiedene Ansätze

zur Modellierung beschrieben. Die Ansätze werden nach bestimmten Kriterien, wie beispielsweise Praktikabilität für Fachmodellierer oder Überführbarkeit ins Systemmodell, bewertet. Ausgehend davon wird ein Ansatz ausgewählt und die Realisierung dieses Ansatzes mit der BPMN 2.0 überprüft. Die Umsetzung der Ansätze zur Modellierung von Bearbeiterzuordnungen und Flexibilitätsstellen ist mit der BPMN 2.0 nicht möglich. Um diese umzusetzen werden Erweiterungen für die BPMN 2.0 vorgestellt.

In Kapitel 5 wird die Modellierung auf der Systemebene betrachtet. Für die betrachteten Aspekte wird gezeigt, wie die im Fachmodell beschriebenen Informationen in das Systemmodell überführt werden können. Anschließend werden die Informationen weiter detailliert, mit dem Ziel einer vollständigen und formalen Modellierung. Danach wird wiederum die Möglichkeit der Umsetzung der Ansätze mit der BPMN 2.0 geprüft.

Soll der Geschäftsprozess durch IT unterstützt werden, muss das Systemmodell zu einer fertigen Applikation weiterentwickelt werden. Das Vorgehen wird in Kapitel 6 vorgestellt. Da der Schwerpunkt dieser Arbeit auf der Modellierung des Fach- und Systemmodells liegt, wird die Weiterentwicklung zu einer fertigen Applikation nur kurz betrachtet. Neben einer allgemeinen Betrachtung wird auf die Möglichkeiten eingegangen, die bei einer Modellierung mit der BPMN 2.0 bestehen.

Um Geschäftsprozesse mit der BPMN modellieren zu können, werden entsprechende Werkzeuge benötigt. In Kapitel 7 werden drei Werkzeuge untersucht, die ARIS Design Platform, der Innovator for Business Analysts sowie IBM WebSphere. Es wird betrachtet wie weit die BPMN 2.0 von diesen Werkzeugen unterstützt wird und ob sich die in Kapitel 4 und 5 vorgestellten Ansätze umsetzen lassen.

In Kapitel 8 werden die Ergebnisse dieser Masterarbeit in Zusammenhang mit anderen Arbeiten gebracht. Dabei werden Erweiterungen der BPMN sowie die Modellierung der Anforderungen auf den unterschiedlichen Ebenen im Hinblick auf das Frontloading betrachtet.

Abschließend lässt sich sagen, dass für alle betrachteten Anforderungen sowohl auf der Fach- als auch auf der Systemebene gute Modellierungsansätze identifiziert wurden. Die Ansätze lassen sich mit der BPMN jedoch nicht vollständig umsetzen. Wo diese Schwächen zeigte, wurden Erweiterungen definiert um diese zu beseitigen.

Ausblick

Die Auseinandersetzung mit der durchgängigen Modellierung von prozessorientierten Anwendungen sowie der Betaversion der BPMN 2.0 haben weitere Fragestellungen aufgeworfen, die Herausforderungen für zukünftige Arbeiten liefern.

Bei der Modellierung von Geschäftsprozessen sollten bereits existierende Informationen früh berücksichtigt werden. Dieser Aspekt wird Look-ahead genannt. Bei der fachlichen Modellierung können etwa bereits existierende Services (Service Look-ahead) oder Daten (Daten Look-ahead) in das Modell integriert werden. In einer weiterführenden Arbeit kann betrachtet werden, wie der Aspekt des Look-ahead die entwickelten Konzepte beeinflusst.

Um bei Änderungen im Geschäftsprozess die Modelle auf den verschiedenen Ebenen konsistent zu halten, ist es nötig die Beziehungen der Elemente zwischen den einzelnen Ebenen zu verwalten.

[BBR10b] definiert dazu ein Abbildungsmodell, dass die Aktivitäten aus dem Fachmodell mit den Aktivitäten des Systemmodells in Verbindung bringt. So können die Modelle bei Änderungen konsistent gehalten werden. Für die weiteren Aspekte (Bearbeiter, Daten, Ausnahmen, Flexibilität) müssen ebenfalls Möglichkeiten zur Konsistenzerhaltung zwischen den Ebenen geschaffen werden.

In dieser Arbeit werden einzelne Aspekte der Geschäftsprozessmodellierung betrachtet. In einem weiteren Schritt muss eine Methode entworfen werden, die die entwickelten Konzepte integriert. Die einzelnen entwickelten Konzepte sowie die Methodik müssen in Fallstudien auf ihre Durchführbarkeit und die Akzeptanz in den Abteilungen praktisch überprüft werden. Dazu müssen für die Erfassung der Informationen durch den Modellierer Oberflächen entwickelt werden. Anschließend können die Konzepte von den Herstellern in die Werkzeuge eingebaut werden.

Literaturverzeichnis

[Aal94] Wil van der Aalst. Putting petri nets to work in industry. *Computers in Industry*, 25(1):45–54, 1994.

[Aal04] Wil van der Aalst. Business Process Management: A Personal View. *Business Process Management Journal*, 10(2):135–139, 2004.

[AB07] Sebastian Herold Christian Linsmeier Detlef Peters Andreas Rausch Alexander Bösl, Jan Ebell. Modellbasierte Software-Entwicklung von Informationssystemen: Vom Geschäftsprozess zum servicebasierten Entwurf. White Paper, 2007.

[AGA+08] Ali Arsanjani, Shuvanker Ghosh, Abdul Allam, Tina Abdollah, Sella Ganapathy, and Kerrie Holley. SOMA: A method for developing service-oriented solutions. *IBM Systems Journal*, 47(3):377–396, 2008.

[Agr07] Ashish Agrawal et al. WS-BPEL Extension for People Specification. Technical report, Active Endpoints, Adobe, BEA, IBM, Oracle, SAP AG, 2007.

[AH05] Wil van der Aalst and Arthur ter Hofstede. YAWL: Yet Another Workflow Language. *Information Systems*, 30(4):245–275, 2005.

[AHKB03] Wil van der Aalst, Arthur ter Hofstede, B. Kiepuszewski, and A. P. Barros. Workflow patterns. *Distributed and Parallel Databases*, 14(1):5–51, July 2003.

[AJ00] Wil van der Aalst and Stefan Jablonski. Dealing with workflow change: identification of issues and solutions. *International Journal of Computer Systems Science and Engineering*, 15(5):267–276, September 2000.

[ALF08] Thomas Henninger Anna-Lena Franck. BPMN 2008. Technical report, camunda services GmbH, 10 2008.

[All07] Thomas Allweyer. Erzeugung detaillierter und ausführbarer Geschäftsprozessmodelle durch Modell-zu-Modell-Transformationen. In Markus Nüttgens, Frank J. Rump, and Andreas Gadatsch, editors, *EPK*, volume 303 of *CEUR Workshop Proceedings*, pages 23–38. CEUR-WS.org, 2007.

[All09] Thomas Allweyer. *BPMN 2.0 - Business Process Model and Notation: Einführung in den Standardard für die Geschäftsprozessmodellierung.* Books on Demand, 2009.

[Ark02] Assaf Arkin et al. Business Process Modeling Language (BPML), Version 1.0, 2002.

[Ars04] Ali Arsanjani. Service-oriented modeling and architecture. IBM developer works, 2004.

[BBP09] Stephan Buchwald, Thomas Bauer, and Rüdiger Pryss. IT-Infrastrukturen für flexible, service-orientierte Anwendungen - ein Rahmenwerk zur Bewertung. In Johann Christoph Freytag, Thomas Ruf, Wolfgang Lehner, and Gottfried Vossen, editors, *BTW*, volume 144 of *LNI*, pages 526–543. GI, 2009.

[BBR10a] Stephan Buchwald, Thomas Bauer, and Manfred Reichert. Durchgängige Modellierung von Geschäftsprozessen durch Einführung eines Abbildungsmodells: Ansätze, Konzepte, Notationen. Technical report, Universität Ulm. Fakultät für Ingenieurwissenschaften und Informatik, April 2010.

[BBR10b] Stephan Buchwald, Thomas Bauer, and Manfred Reichert. Durchgängige Modellierung von Geschäftsprozessen in einer Service-orientierten Architektur. In *Modellierung'10*, Lecture Notes in Informatics (LNI). Koellen-Verlag, March 2010.

[BFF+08] Pascal Bauler, Fernand Feltz, Etienne Frogneux, Benjamin Renwart, and Céline Thomase. Usage of model driven engineering in the context of business process management. In Martin Bichler, Thomas Hess, Helmut Krcmar, Ulrike Lechner, Florian Matthes, Arnold Picot, Benjamin Speitkamp, and Petra Wolf, editors, *MKWI*. GITO-Verlag, Berlin, 2008.

[BP05] Ross Brown and Helen Paik. Resource-centric worklist visualisation. January 2005.

[Bro04] Alan W. Brown. Model driven architecture: Principles and practice. *Software and System Modeling*, 3(4):314–327, 2004.

[Che08] Hong-Mei Chen. Towards Service Engineering: Service Orientation and Business-IT Alignment. *Hawaii International Conference on System Sciences*, page 114, 2008.

[CS94] Rudong Chen and August-Wilhelm Scheer. Modellierung von Prozessketten mittels Petri-Netz Theorie. Veröffentlichungen des Instituts für Wirtschaftsinformatik, Heft 107 (in German), University of Saarland, Saarbrücken, 1994.

[DPRW08] Andreas Ditze, Detlef Peters, Gerhard Rempp, and Lutz Wiegmann. Modellbasiert zum Ziel: MID ModellierungsMethodik M3. *Modeling Magazin*, pages 4–5, October 2008.

[DS08] Gero Decker and Torben Schreiter. OMG releases BPMN 1.1 - what's changed? *EMISA Forum*, 28(2):12–20, 2008.

[End09] Ralf Enderle. Frühe fachliche Modellierung ausführungsrelevanter Prozess-Aspekte - Prozessmodellierung in Zeiten von SOA. Master's thesis, Universität Ulm, October 2009.

[Erl05] Thomas Erl. *Service-Oriented Architecture (SOA): Concepts, Technology, and Design*. Prentice Hall, August 2005.

[FRH10] Jakob Freund, Bernd Rücker, and Thomas Henninger. *Praxishandbuch BPMN*. Hanser Fachbuch, 2010.

[Gro07] Alexander Grosskopf. xBPMN - Formal Control Flow Specification of a BPMN based Process Execution Language. Master's thesis, Hasso Plattner Institute for IT Systems Engineering, Potsdam, Germany, 2007.

[IBM08] IBM. WebSphere Process Server, Version 6.2. White Paper, 2008.

[IBM09] IBM. Rational Software. http://www.ibm.com/software/rational, December 2009.

[IBM10] IBM. IBM Software - WebSphere. http://www.ibm.com/websphere, April 2010.

[IDS05] IDS Scheer. Business Process Management: ARIS Value Engineering-Concept. Whitepaper, 2005.

[IDS10a] IDS Scheer. ARIS Platform. Technical report, ARIS Delta Paper, February 2010.

[IDS10b] IDS Scheer. Business Process Excellence: The Benchmark for Enterprise-Wide Process Design. http://www.ids-scheer.com/en/ARIS/ARIS_Platform/ARIS_Design_Platform/6926.html, May 2010.

[IDS10c] IDS Scheer. *Modeling BPMN 2.0 in ARIS 7.1*, January 2010.

[Jab97] Stefan Jablonski. Architektur von Workflow-Management-Systemen. *Informatik Forschung und Entwicklung, Themenheft Workflow-Management*, 12(2):72–81, 1997.

[Jen87] Kurt Jensen. Coloured petri nets. pages 248–299. 1987.

[JKJ+04] Moonyoung Jung, Hak Soo Kim, Myung Hyun Jo, Kyung Hyun Tak, Hyun Suk Cha, and Jin Hyun Son. Mapping from BPMN-Formed Business Processes to XPDL Business Processes. In Jian Chen, editor, *ICEB*, pages 422–427. Academic Publishers/World Publishing Corporation, 2004.

[Jos07] Nicolai M. Josuttis. *SOA in Practice: The Art of Distributed System Design*. O'Reilly Media, 1 edition, August 2007.

[KLL09] Ryan K. L. Ko, Stephen S. G. Lee, and Eng Wah Lee. Business process management (BPM) standards: a survey. *Business Process Management Journal*, 15(5):744–791, 2009.

[KNS92] Gerhard Keller, Markus Nüttgens, and August-Wilhelm Scheer. Semantische Processmodellierung auf der Grundlage Ereignisgesteuerter Processketten (EPK). Institut für Wirtschaftsinformatik, Heft 89, University of Saarland, Saarbrücken, 1992.

[LÖ09] Ling Liu and M. Tamer Özsu. ACID transaction. In *Encyclopedia of Database Systems*, pages 21–26. Springer US, 2009.

[MID10a] MID. Innovator for Business Analysts. Technical report, Data Sheet, 2010.

[MID10b] MID. Modellierungsplattform Innovator. http://mid.de/Modellierungsplattform-Innovat.innovator.0.html, April 2010.

[MR08] Michael zur Muehlen and Jan Recker. How much language is enough? Theoretical and practical use of the business process modeling notation. In Zohra Bellahsène and Michel Léonard, editors, *Proceedings of the 20th Int'l Conference on Advanced Information Systems Engineering (CAiSE 2008)*, pages 465–479, Montpellier, France, 2008. Springer Verlag.

[OAS05] OASIS. *Web Services Transaction*, 1.2 edition, August 2005.

[OAS07] OASIS. *Web Services Business Process Execution Language Version 2.0*, april 2007.

[ODBH05] Chun Ouyang, Marlon Dumas, Stephan W. Breutel, and Arthur ter Hofstede. Translating standard process models to BPEL. December 2005.

[ODHA06] Chun Ouyang, Marlon Dumas, Arthur ter Hofstede, and Wil van der Aalst. From BPMN process models to BPEL web services. January 2006.

[OMG03] OMG. *MDA Guide Version 1.0.1.* Framingham, Massachusetts, June 2003.

[OMG06] OMG. Object constraint language specification, version 2.0. http://www.omg.org/technology/documents/formal/ocl.htm, May 2006.

[OMG09a] OMG. *Business Process Model and Notation (BPMN)*, version 1.2 edition, January 2009.

[OMG09b] OMG. *Business Process Model and Notation (BPMN)*, FTF Beta 1 for Version 2.0 edition, September 2009.

[OMG09c] OMG. *UML 2.2 Superstructure Specification*, February 2009.

[Pap03] Michael P. Papazoglou. Web Services and Business Transactions. Technical report, January 2003.

[Pei09] Roland Peisl. Business Process Management mit IBM WebSphere. Whitepaper, IBM, November 2009.

[Pet62] Carl Adam Petri. *Kommunikation mit Automaten.* PhD thesis, TU Darmstadt, Bonn, Germany, 1962.

[RCWR04] Manfred Reichert, Ursula Catrinescu-Wiedemuth, and Stefanie Rinderle. Evolution von Zugriffsregelungen in Informationssystemen. In *Proc. Conf. eBusiness Processes (EBP'04)*, pages 100–114, September 2004.

[Rei00] Manfred Reichert. *Dynamische Ablaufänderungen in Workflow-Management-Systemen.* PhD thesis, Universität Ulm, Fakultät für Informatik, 2000.

[Rup07] Chris Rupp. *Requirements-Engineering und -Management.* Hanser Fachbuch, 2007.

[Sch92] August-Wilhelm Scheer. *Architektur Integrierter Informations-Systeme: Grundlagen Der Unternehmensmodellierung.* Springer, 1992.

[Sch04] Rolf Scheuch. Mit der UML 2.0 Geschäftsprozesse modellieren: Ein Vergleich marktrelevanter Methoden und Alternativen. *OBJEKTspektrum*, 3:30–33, 2004.

[Sch08] David Schumm. Graphische Modellierung von BPEL Prozessen unter Verwendung der BPMN Notation. Master's thesis, Universität Stuttgart, 2008.

[Sei06] Sinje Seidler. Vom Geschäftsprozess in die SAP Konfiguration. Technical report, ARIS Expert Paper, 2006.

[SH09] Sebastian Stein and Erik Hagen. Adopting BPMN with ARIS. Technical report, ARIS Expert Paper, Saarbrücken, Germany, January 2009.

[Sha02] Robert Shapiroro. A comparison of XPDL, BPML and BPEL4WS (version 1.4). http://xml.coverpages.org/Shapiro-XPDL.pdf, 2002.

[SI07] Sebastian Stein and Konstantin Ivanov. EPK nach BPEL Transformation als Voraussetzung für praktische Umsetzung einer SOA. In Wolf-Gideon Bleek, Jörg Raasch, and Heinz Züllighoven, editors, *Software Engineering*, volume 105 of *LNI*, pages 75–82. GI, 2007.

[Sil09] Bruce Silver. *BPMN Method and Style: A levels-based methodology for BPM process modeling and improvement using BPMN 2.0.* Cody-Cassidy Press, June 2009.

[Ste09] Sebastian Stein. *Modelling Method Extension for Service-Oriented Business Process Management.* PhD thesis, Christian-Albrechts-Universität zu Kiel, Kiel, Germany, December 2009.

[TNB06] Jan Trautvetter, Aydin Necati, and Felix Billau. Vergleich von kommerziellen Implementierungen eines Enterprise Service Bus. Technical report, Universität Stuttgart; Fakultät Informatik, Elektrotechnik und Informationstechnik. Institut für Architektur von Anwendungssystemen, 2006.

[VVLM08] Jussi Vanhatalo, Hagen Völzer, Frank Leymann, and Simon Moser. Automatic workflow graph refactoring and completion. pages 100–115. 2008.

[W3C07] W3C. XML Path Language (XPath) 2.0. http://www.w3.org/TR/xpath20/, January 2007.

[Wes07] Mathias Weske. *Business Process Management: Concepts, Languages, Architectures.* Springer, 1 edition, November 2007.

[WFM02] WFMC. Workflow management coalition workflow standard: Workflow process definition interface – XML process definition language (XPDL) (WFMC-TC-1025). Technical report, Workflow Management Coalition, Lighthouse Point, Florida, USA, 2002.

[Whi05] Stephan White. Using BPMN to model a BPEL process. *BPTrends*, 3(3):1–18, March 2005.

[ZD08] Man Zhang and Zhenhua Duan. From business process models to web services orchestration: The case of UML 2.0 activity diagram to BPEL. In Athman Bouguettaya, Ingolf Krüger, and Tiziana Margaria, editors, *ICSOC*, volume 5364 of *Lecture Notes in Computer Science*, pages 505–510, 2008.

A

Events in der BPMN

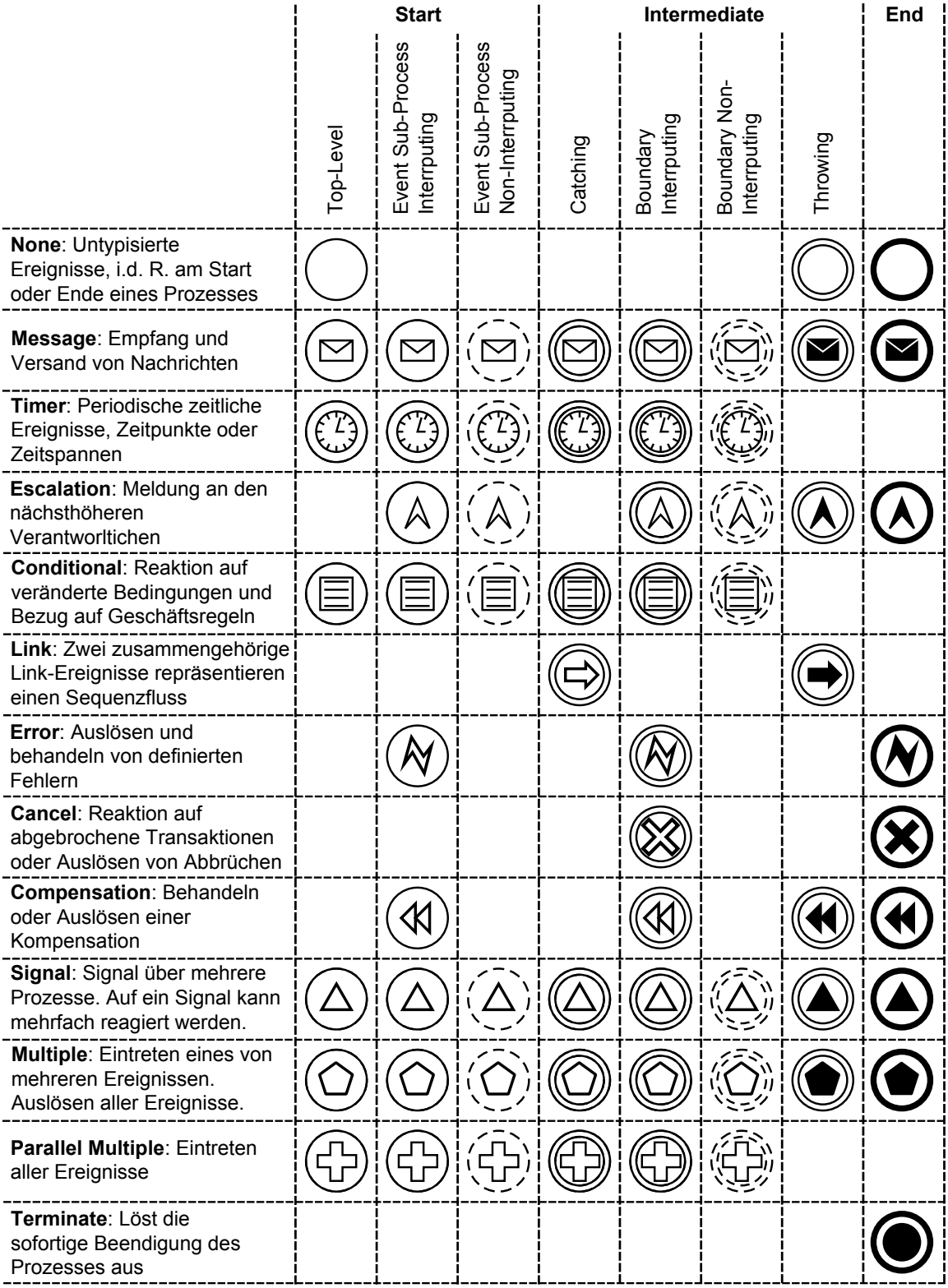


Abbildung A.1: Tabellarische Übersicht der BPMN-Events

B

Flexibilität

Definition eines neuen Elements zur Markierung von Flexibiliätsstellen (analog zu [OMG09b]):

Extended Artifacts Metamodell

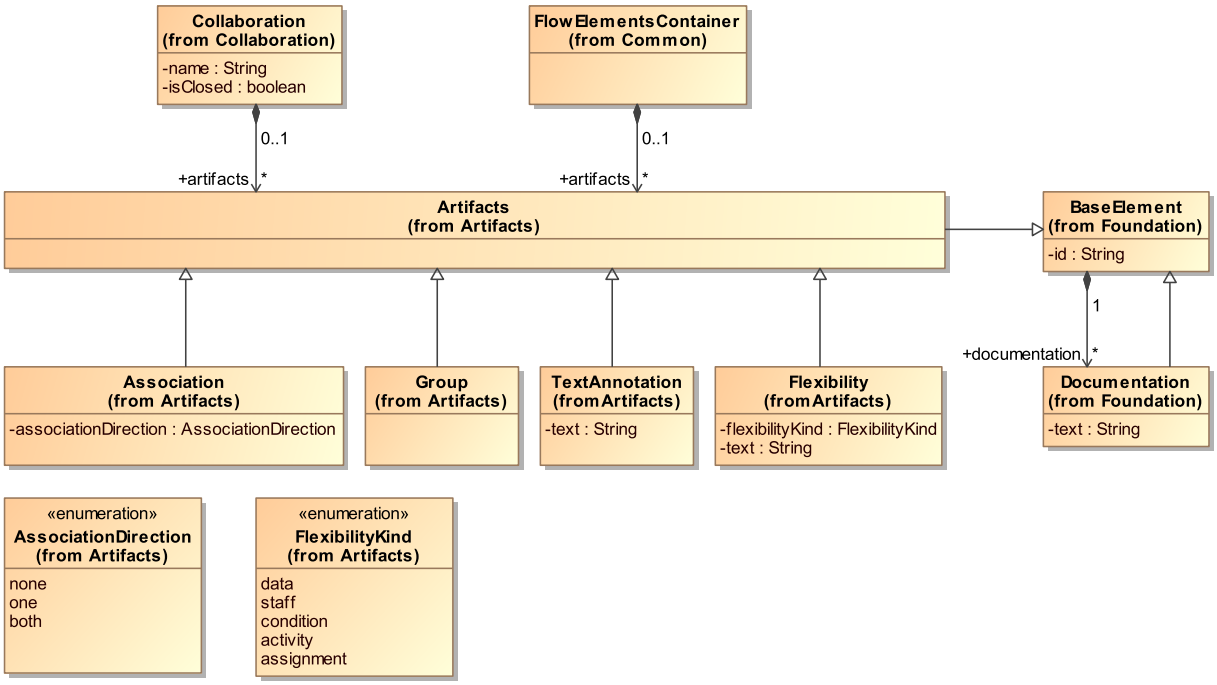


Abbildung B.1: Extended Artifacts Metamodell

Flexibility Attributes

Attributes	Description
flexibilityKind:FlexibilityKind data | staff | condition | activity | assignment	Defines the nature of the flexibility Possible values are data, staff, condition, activity or assignment
text:string	This attribute is used to describe in detail the kind of flexibility

Flexibility XML Schema

```
01 <xsd:element name="flexibility" type="tFlexibility" ➲
02 substitutionGroup="artifact"/>
03 <xsd:complexType name="tFlexibility">
04   <xsd:complexContent>
05     <xsd:extension base="tArtifact">
06       <xsd:sequence>
07         <xsd:element name="text" type="xsd:string" minOccurs="0" ➲
08         maxOccurs="1"/>
09       </xsd:sequence>
10       <xsd:attribute name="flexibilityKind" type="tFlexibilityKind" ➲
11       use="required"/>
12     </xsd:extension>
13   </xsd:complexContent>
14 </xsd:complexType>
15
16 <xsd:simpleType name="tFlexibilityKind">
17   <xsd:restriction base="xsd:string">
18     <xsd:enumeration value="data"/>
19     <xsd:enumeration value="staff"/>
20     <xsd:enumeration value="condition"/>
21     <xsd:enumeration value="activity"/>
22     <xsd:enumeration value="assignment"/>
23   </xsd:restriction>
24 </xsd:simpleType>
```

C
Glossar

.NET .NET ist ein Framework zur Softwareentwicklung von Microsoft. Das Framework vereinigt die Enwicklungsparadigmen Objektorientierung, Komponentenorientierung und Serviceorientierung und ist programmiersprachenneutral.

ACID ACID ist ein in der Informatik gebräuchliches Akronym und steht für atomicity, consistency, isolation und durability (auf deutsch atomar, konsistent, isoliert und dauerhaft). Diese Eigenschaften sollte eine Transaktion erfüllen.

ausführbares Modell Modell eines Geschäftsprozesses, welches aus dem Systemmodell entwickelt wird. Das ausführbare Modell ist eine ausführbare Applikation des modellierten Geschäftsprozesses und unterstützt beziehungsweise automatisiert diesen.

BPEL Die Business Process Execution Language (BPEL) ist eine XML basierte Sprache zur Beschreibung von Geschäftsprozessen. Die mittels BPEL modellierten Geschäftsprozesse können durch eine Process-Engine ausgeführt werden.

BPM Das Business Process Management (BPM) (im Deutsch Geschäftsprozessmanagement) beschäftigt sich mit dem Identifizieren, Modellieren, Analysieren, Optimieren, Ausführen und Kontrollieren von Geschäftsprozessen.

BPMI Die Business Process Management Initiative (BPMI) war eine Konsortium von mehr als 90 Unternehmen. Das Konsortium hatte sich zum Ziel gesetzt einen ganzheitlichen offenen Standard für die Modellierung, Implementierung und Ausführung von Geschäftsprozessen zu entwickeln. Im Juni 2005 fusionierte die BPMI mit der OMG.

BPML Die Business Process Modeling Language (BPML) wird dazu verwendet Prozessbeschreibungen zu spezifizieren, die mit einer Process-Engine ausgeführt werden können. Diese Aufgabe wurde jedoch mehr und mehr von BPEL übernommen, weshalb die Entwicklung der BPML nicht mehr fortgeführt wurde.

BPMN Die Business Process Modeling Notation (BPMN) ist eine Notation, mit der Geschäftsprozesse modelliert werden können. Derzeit wird an einer neuen Version 2.0 gearbeitet, die aktuell im Betastadium vorliegt. In der neuen Version steht das Akronym BPMN für Business Process Model and Notation. Die BPMN eignet sich sowohl für die Modellierung auf Fachebene als auch für die Modellierung auf Systemebene.

BRMS Ein Business-Rule-Management-System (BRMS) verwaltet Geschäftsregeln. Diese können von externen Anwendungen aufgerufen werden. Nach der Übergabe von Parametern erfolgt die Berechnung der Geschäftsregel im BRMS. Das Ergebnis wird anschließend zurückgegeben.

Business-IT-Alignment Wechselseitige Abstimmung von Zielen, Strategien und Leistungen zwischen IT- und Fachabteilung in einem Unternehmen.

CORBA Die Common Object Request Broker Architecture (CORBA) wurde von der OMG entwickelt. Sie ist ein Modell für die Entwicklung verteilter objektorientierter Anwendungen.

EPK Eine Ereignisgesteuerte Prozesskett (EPK) ist eine Notation zur Geschäftsprozessmodellierung. Aufgrund ihrer Einfachheit eignet sie sich insbesondere für den Einsatz in den Fachabteilungen. Für eine formale Modellierung der Geschäftsprozesse, wie sie im Systemmodell benötigt wird, ist sie nicht geeignet.

Fachmodell Modell eines Geschäftsprozesses, welches von einer Fachabteilung erstellt wird.

Fachmodellierer Modellierer des Fachmodells. Meist Mitarbeiter der entsprechenden Fachabteilung.

Frontloading Frühes Modellieren von ausführungsrelevanten Aspekten, um spätere Abstimmungsrunden zwischen IT und Fachbereich zu vermeiden.

Geschäftsprozess Abfolge von Schritten zur Erreichung eines geschäftlichen Zieles.

Geschäftsprozessmodellierung Ein Geschäftsprozess wird dargestellt. Dies geschieht meist grafisch. Der Schwerpunkt der Modellierung liegt auf dem Ablauf, darüber hinaus werden jedoch auch weitere Informationen wie Daten oder Bearbeiter modelliert. Die Geschäftsprozessmodellierung ist Teil des Geschäftsprozessmanagements.

J2EE Die Java Platform, Enterprise Edition (J2EE) ist die Spezifikation einer Softwarearchitektur für die transaktionsbasierte Ausführung von in Java porgrammierten Anwendungen. Sie ist wie auch .NET im Bereich der Middleware anzusiedeln.

MDA Model Driven Architecture (MDA) bezeichnet einen modellgetriebenen Softwareentwicklungsansatz. Dabei wird auf eine klare Trennung von Funktionalität und Technik geachtet.

OMG Die Object Management Group (OMG) ist ein Konsortium, das sich mit der Entwicklung von Standards beschäftigt. Der 1989 gegründeten Vereinigung gehören mittlerweile mehr als 800 Mitglieder an, darunter zahlreiche namhafte wie etwa IBM und Microsoft.

Petri-Netz Ein Petri-Netz ist ein mathematisches Modell mit dem man Geschäftsprozesse modellieren kann. Aufgrund der exakten mathematischen Definition der Ausführungssemantik, eignen sich Petri-Netze gut für Prozessanalysen.

Process-Engine Eine Anwendung zur Ausführung von Prozessen. Die auszuführenden Prozesse müssen zuvor in einer Ausführungssprache definiert werden.

SOA In einer serviceorientierte Architektur (SOA) werden Geschäftsprozesse mittels lose gekoppelten Einheiten (Services) abgebildet. Ziel dieser Entkopplung ist es, die Flexibilität zu erhöhen.

Systemmodell Modell eines Geschäftsprozesses, welches durch die Weiterentwicklung des Fachmodells entsteht. Es wird durch die IT-Abteilung in Abstimmung mit der Fachabteilung erstellt und bildet den modellierten Geschäftsprozess vollständig und formal ab.

Systemmodellierer Modellierer des Systemmodells. Mitarbeiter in der IT-Abteilung.

Transaktion Folge von Operationen, die als logische Einheit betrachtet werden.

UML Die Unified Modeling Language (UML) ist ein von der OMG entwickelter Standard für die Modellierung von Software und anderen Systemen.

UML Activity Chart Das Activity Chart ist ein Diagrammart aus dem Standard der UML. Mit Activity Charts können Geschäftsprozesse modelliert werden. Die Notation eignet sich aufgrund ihres Umfangs und der Formalität insbesondere für die IT, ist für Fachabteilungen jedoch schwer verständlich.

XMI XML Metadata Interchange (XMI) ist ein auf XML basierendes Austauschformat zwischen verschiedenen Werkzeugen.

XML Die Extensible Markup Language (XML) ist eine Auszeichnungssprache zur Darstellung von hierarchisch sturkturierten Daten in Textform.

XPath Die XML Path Language (XPath) ist eine Abfragesprache für XML Dokumente. Mit ihr lassen sich Teile eines XML-Dokumentes adressieren.

XPDL Die XML Process Definition Language (XPDL) ist eine XML basierte Sprache zur Beschreibung von Geschäftsprozessen. Die mit ihr definierten Geschäftsprozesse sind automatisch ausführbar, XPDL ist folglich eine Ausführungssprache.