

Kapitel 2

Die Notation im Detail

2.1 BPMN verstehen

Was weiß ein Affe vom Geschmack von Ingwer?

Die diesem indischen Sprichwort zugrunde liegende Erkenntnis lautet: „Jemand, der etwas nicht verstehen kann, ist nicht in der Lage, es zu schätzen.“ Ein heimisches Äquivalent wäre vermutlich: „Perlen vor die Säue werfen“.

Auch die BPMN ist eine Perle, die nicht jeder zu schätzen weiß – weil sie nicht jeder versteht. Deshalb bitten wir Sie darum, sich ein wenig Zeit zu nehmen, um sich mit den grundsätzlichen Prinzipien dieses Standards vertraut zu machen. Sie werden es nicht bereuen. Wenn Sie die BPMN wirklich verstanden haben, werden Sie ein ausgesprochen mächtiges Werkzeug gewinnen, das Ihnen in modernen BPM-Projekten von großem Nutzen sein wird.

Wenn Sie die BPMN-Spezifikation bereits kennen, werden Sie in diesem Kapitel nicht viel Neues erfahren. Im Grunde handelt es sich „nur“ um eine leichter verdauliche und natürlich deutschsprachige Fassung. Zusätzlich geben wir ein paar erläuternde Hinweise, die Sie in der Spezifikation nicht finden werden, und beschreiben die visuellen Konventionen, die wir bei der Anwendung der Symbole verwenden (unser „BPMN-Knigge“).

Aber auch wenn Sie glauben, dass Sie die BPMN bereits kennen oder sie sogar schon angewandt haben: Lesen Sie wenigstens diesen Abschnitt. Die Erfahrung hat uns gelehrt, dass viele, die meinen, die BPMN schon sehr gut zu kennen, die wichtigsten Basisprinzipien der Notation noch gar nicht verstanden haben – und sich immer noch wundern, dass man „Sequenzflüsse“ nicht über „Pool-Grenzen“ hinweg malen darf.

2.1.2 Eine Landkarte: Die BPMN-Kernelemente

Prozessdiagramme, die mit der BPMN erstellt wurden, heißen „Business Process Diagrams (BPD)“. Wann immer Sie ein BPD zeichnen, verwenden Sie Symbole, die sich den in Abbildung 2.1 dargestellten Kategorien zuordnen lassen. Diese Kategorien heißen deshalb auch „BPMN-Kernelemente“.

Im Prinzip müssen in einem Prozess bestimmte Dinge getan werden (*Aktivitäten*), möglicherweise aber nur unter bestimmten Bedingungen (*Gateways*), und es können Dinge passieren (*Ereignisse*). Diese drei Flussobjekte werden über *Sequenzflüsse* miteinander verbunden, jedoch nur innerhalb eines *Pools* bzw. einer *Lane*. Falls eine Verbindung über Poolgrenzen hinweg erfolgt, greift man zu den *Nachrichtenflüssen*. Zu guter Letzt gibt es *Artefakte*, die weitere Informationen zum Prozess liefern sollen, aber keinen direkten Einfluss auf die Reihenfolge der Flussobjekte haben können. Jedes Artefakt kann prinzipiell mit jedem Flussobjekt verbunden werden, und zwar mit Hilfe von *Assoziationen*.

Das war's eigentlich schon – jetzt kennen Sie die BPMN. Im Entwurf zur BPMN 2.0 werden die Datenobjekte zwar nicht mehr als Artefakte behandelt, sondern in eine eigene Kategorie gehoben. Ansonsten wäre dies aber das Basis-Schema, nach dem die BPMN funktioniert.

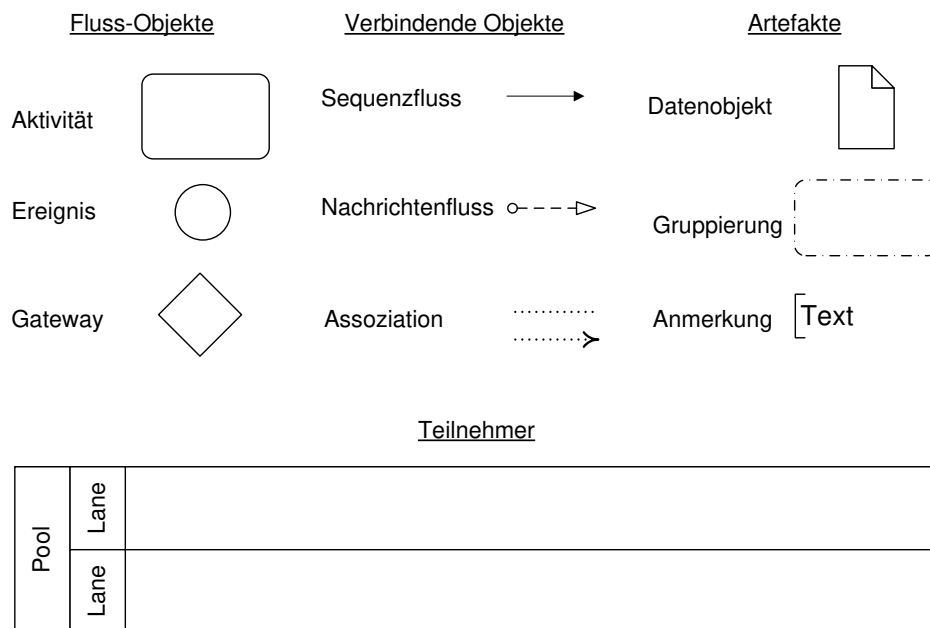


Abbildung 2.1: Die Kernelemente der BPMN

Nun ja, wenn wir ehrlich sind, fehlen noch drei winzige Aspekte für ein echtes Verständnis der BPMN:

- die weiterführenden Gedanken und Regeln, die sich hinter diesem simplen Schema verbergen;
- die vollständige Palette der Symbole;
- die Frage, was man mit dem ganzen Zeugs in der Praxis *wirklich* anfängt.

Die ersten beiden Punkte werden wir in diesem Kapitel klären. Der dritte Punkt bezieht sich auf einen „Soft-Skill“, den man eigentlich nur durch eigene Erfahrung aufbauen kann. Um diesen Prozess für Sie etwas zu beschleunigen, haben wir in den weiteren Kapiteln unsere Erfahrungen zusammengetragen und versucht, daraus ein paar „Kochrezepte“ bei der praktischen Anwendung von BPMN abzuleiten. Vielleicht helfen sie Ihnen dabei, nicht in alle Fallen zu tappen, in denen wir uns anfangs verirrt haben.

2.1.3 Perspektiven bei der Prozessbetrachtung

Wer bereits Prozesse mit anderen Notationen modelliert hat, kann sich an einen extrem wichtigen Aspekt von BPMN häufig nur schwer gewöhnen: Alles ist eine Frage der Perspektive.

BPMN geht davon aus, dass es in einem BPD einen oder mehrere *Teilnehmer* (engl.: participant) geben kann. Gehen Sie mit diesem Begriff sehr vorsichtig um, und setzen Sie ihn beispielsweise nicht vorschnell mit „Rolle“, „Abteilung“ oder „Mitarbeiter“ gleich! Ein „Teilnehmer“ ist für BPMN zunächst einmal ein rein logisches Element, für das die folgenden Regeln gelten:

- Für einen Prozess existiert nur ein einziger Teilnehmer (ja, das ist erst mal verwirrend).
- Dieser Teilnehmer hat die totale Kontrolle über den Prozessfluss.
- Andere Teilnehmer können den Prozess dieses Teilnehmers nicht beeinflussen, unter Umständen wissen sie nicht einmal, wie er funktioniert.
- Der Teilnehmer ist dementsprechend für diesen Prozess verantwortlich.
- Wenn der Teilnehmer im Rahmen seines Prozesses mit anderen Teilnehmern interagieren möchte, muss er mit ihnen Nachrichten austauschen, was diese wiederum in ihren eigenen Prozessen entsprechend unterstützen müssen.

Derselbe Prozess kann deshalb für die jeweiligen Teilnehmer ganz unterschiedlich aussehen, eben abhängig von ihrer jeweiligen Perspektive. Das führt automatisch zu unterschiedlichen Prozessmodellen.

Das BPMN-Symbol für den Prozess eines Teilnehmers ist der Pool. Er entspricht gleichzeitig dem Teilnehmer selbst, aber inhaltlich gesehen könnte ein Teilnehmer natürlich mehr als nur einen Prozess steuern.

Wenn Sie lernen, mit Pools richtig umzugehen, haben Sie das vielleicht wichtigste Prinzip der Prozessmodellierung überhaupt verstanden – zumindest, wenn Sie ein modernes Business Process Management mit dem notwendigen Business-IT-Alignment anstreben. Wir werden uns diesem Thema in Abschnitt 2.9 auf Seite 95 widmen und auch das Rätsel lösen, warum es für einen Prozess nur einen einzigen Teilnehmer im Sinne der BPMN, aber durchaus mehrere Teilnehmer im herkömmlichen Sinne geben kann.

2.1.4 Modelle, Instanzen, Token und Korrelationen

Im Entwurf zur BPMN 2.0 finden Sie im 7. Kapitel „Überblick“ einen Abschnitt, den es in der aktuellen Fassung 1.2 noch nicht gibt: „Das Verhalten von Diagrammen verstehen“. Gemeint ist damit, dass Sie das Verhalten der Prozesse verstehen müssen, die in den Diagrammen beschrieben sind (Anmerkung: Da in einem Business Process Diagram durchaus mehrere Pools enthalten sein können, gilt unter Umständen: 1 Diagramm – n Prozesse). Das ist ein sinnvoller Anspruch, aber häufig leichter gesagt, als getan: Manche Prozessmodelle sind so komplex, dass es für den Betrachter schwierig sein kann, genau nachvollzuziehen, unter welchen Umständen welche Dinge zu tun sind. Es wird jedoch erheblich einfacher, wenn Sie die folgenden Begriffe und Prinzipien verinnerlichen:

- **Prozessmodell:** In einem BPD sind ein oder mehrere Prozessmodelle beschrieben. Das Modell ist die prinzipielle Beschreibung des Prozesses.
- **Prozessinstanz:** Wenn der Prozess in der Realität durchlaufen wird, handelt es sich um eine Prozessinstanz. Für Laien könnte man dies auch als einen „Vorgang“ bezeichnen. Die eingehende Beschwerde eines Kunden beispielsweise erzeugt eine Instanz des Reklamationsprozesses. Manche Prozesse werden vielleicht nur wenige Male pro Jahr instanziiert, zum Beispiel der buchhalterische Monatsabschluss. Andere hingegen etwas häufiger: Der Auskunftsprozess, den die Schufa betreibt, wird laut Website des Unternehmens jährlich ca. 90 Mio. mal instanziiert.
- **Token:** Wenn man ein Prozessmodell vor Augen hat und sich vorstellen möchte, welche Prozesspfade während einer Prozessinstanz zwingend oder möglicherweise durchlaufen werden, kann man das Token-Konzept anwenden. Das Token ist ein theoretisches Konstrukt, das man mit einem Auto vergleichen könnte: Das Auto folgt dem Straßenverlauf. Wenn es an eine Kreuzung kommt, muss sich der Fahrer entscheiden, ob er abbiegen oder geradeaus fahren möchte. Es kann sogar passieren, dass das Auto an einen Punkt kommt, wo es abbiegen *und* gerade ausfahren soll, dann muss es „geklont“ werden. OK, das ist vielleicht etwas unrealistisch, aber Sie haben hoffentlich bereits gemerkt, worum es geht: Das Straßennetz ist das Prozessmodell, und der Weg des Autos entspricht den konkreten Prozesspfaden, die im Rahmen der Instanz durchlaufen werden. Wenn Sie das Token-Konzept anwenden, werden Sie jedes BPMN-Prozessmodell verstehen können, und sei es noch so komplex.

Wir werden im Verlauf dieses Buches diese Methode oft anwenden, um unsere Beispiele durchzusprechen.

- **Korrelation:** Bekommen Sie auch manchmal Briefe von Behörden oder Firmen, die eine Vorgangsnummer oder ein Aktenzeichen enthalten? Wenn Sie auf diese Briefe antworten, Einspruch erheben wollen oder Ähnliches, müssen Sie stets dieses Kennzeichen angeben. Ansonsten kann die Gegenseite Ihre Nachricht nicht zuordnen. Diese Zuordnung auf Basis eines eindeutigen Schlüssels nennt man Korrelation. Ein anderes Beispiel ist eine Rechnung mit dem Hinweis, dass Sie bei der Überweisung die Rechnungsnummer im Verwendungszweck angeben müssen. Wenn das nicht passiert, kann Ihre Zahlung nicht zugeordnet, also korreliert, werden, was zu einer kostenpflichtigen Mahnung führen kann. Das Thema Korrelation ist sowohl für die organisatorische als auch die technische Gestaltung von Prozessen häufig erfolgskritisch, und leider auch häufig ein Bereich, in dem durch Unachtsamkeiten sehr teure Fehler gemacht werden.

2.1.5 BPMN auf Deutsch

Bei einer unserer ersten BPMN-Schulungen ist uns Folgendes passiert: Die Teilnehmer der Schulung kamen nicht aus der IT, sondern der Betriebsorganisation. Sie interessierten sich also für die rein fachliche Prozessdokumentation mit Hilfe der BPMN und erwogen eine Ablösung ihrer alten Notation (EPK). Damals benutzten wir in der Schulung noch die englischen Bezeichnungen für die BPMN-Symbole, wie sie auch in der Spezifikation zu finden sind. Als wir bei den Error-Events angekommen waren und gerade davon sprachen, dass man diese „catchen“ könnte, um ein „error handling“ durchzuführen und den error gegebenenfalls wieder zu „thrown“, wurden wir von einem der Teilnehmer unterbrochen:

Was faseln Sie denn da? Error? Catchen? Thrown? Das kenne ich doch von unseren Programmierern, die reden dauernd von sowas. Aber wir sind doch keine Techies, wir sind Organisatoren! So einen Kram brauchen wir nicht!

Zustimmendes Gemurmel erhob sich, und wir konnten die Gruppe nur mit Mühe und Geduld davon überzeugen, dass die vorgestellten Symbole auch für eine rein organisatorische Prozessbetrachtung durchaus hilfreich sein können. Bei der nächsten Schulung mit ähnlichen Teilnehmern führten wir ein kleines Experiment durch: Wir erläuterten haargenau denselben Sachverhalt, verwendeten aber stattdessen deutsche Begriffe. Es ging also jetzt darum, dass es bei der Prozessdurchführung zu „Fehlern“ kommen kann, die rechtzeitig „erkannt“ und „behoben“ bzw. im Notfall an eine höhere Stelle „gemeldet“ werden müssen. Da nickten die anwesenden Organisatoren zustimmend, denn „Fehler passieren bei uns auch. Das müssen wir berücksichtigen“.

Danach setzten wir uns mit ein paar befreundeten Firmen und Hochschulen zusammen und entwarfen eine deutsche Übersetzung der BPMN-Symbole. Die damals definierten Begriffe verwenden wir bis heute.

Zum Nachschlagen finden Sie die Übersetzung der BPMN-Symbole als Tabelle im Anhang dieses Buches.

2.1.6 Symbole und Attribute

Die BPMN-Spezifikation beschreibt nicht nur die für die Prozessmodellierung verfügbaren Symbole, sondern auch eine ganze Reihe von Attributen, die man an den Symbolen zusätzlich hinterlegen kann. Ein guter Teil dieser Attribute wird im Diagramm nicht visualisiert, sondern nur in dem Tool gespeichert, mit dem Sie modellieren. Das liegt vor allem daran, dass man die in BPMN erstellten Modelle unter Umständen in eine Prozess-Ausführungssprache übersetzen lassen will, damit eine Process Engine auf dieser Grundlage den Prozess automatisieren kann. Im Entwurf zur BPMN 2.0 ist sogar vorgesehen, die in BPMN erstellten Modelle direkt durch die Process Engine ausführen zu lassen.

2.2 Einfache Aufgaben und Blanko-Ereignisse

Abbildung 2.2 zeigt einen sehr einfachen Prozess. Er wird durch die Tatsache ausgelöst, dass jemand hungrig ist. In der Folge müssen Lebensmittel eingekauft und eine Mahlzeit zubereitet werden. Am Ende wird die Mahlzeit verzehrt, und der Hunger ist gestillt.

In diesem BPD kann man die folgenden Symbole und ihre Bedeutung leicht erkennen:

Aufgaben

Die Aufgaben sind das „Herz“ des Prozesses. Schließlich geht es vor allem darum, dass irgendetwas getan werden muss, damit der Prozess die gewünschte Leistung erbringen kann. Streng genommen gehört eine Aufgabe in BPMN zur Kategorie

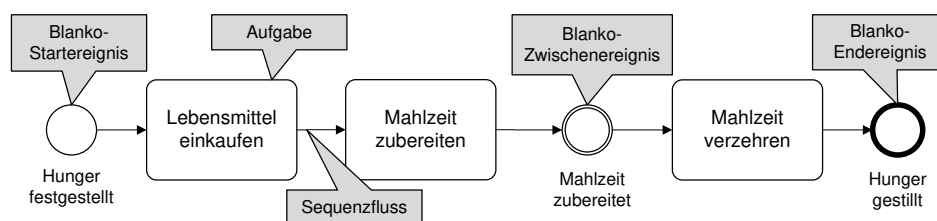


Abbildung 2.2: Unser erster Prozess

der Aktivitäten, zu der auch die in Abschnitt 2.8 auf Seite 80 erklärten Teilprozesse gehören.

Unser BPMN-Knigge

Wenn wir Aufgaben bezeichnen, versuchen wir das Objekt-Verrichtungsprinzip einzuhalten. Das bedeutet, wir bezeichnen sie immer mit dem [Objekt] + [Verb] – Pattern, also beispielsweise „Lebensmittel einkaufen“, und nicht „Zuerst muss der Einkauf der Lebensmittel erledigt werden“.

Ereignisse

Ereignisse stellen dar, dass vor, während oder am Ende des Prozesses etwas Betrachtenswertes passiert. In diesem Beispiel arbeiten wir nur mit sogenannten „Blanko-Ereignissen“, später lernen wir weitere Arten von Ereignissen kennen.

- Startereignisse zeigen, welches Ereignis dazu führt, dass der Prozess gestartet wird.
- Zwischenereignisse stehen für einen Status, der im Prozess erreicht wird und den man im Modell explizit festhalten möchte. Sie werden eher selten benutzt, können aber sehr nützlich sein – zum Beispiel, wenn man den Status als Meilenstein versteht und die Zeit bis zur Erreichung des Meilensteines messen möchte.
- Endereignisse kennzeichnen den Status, der am Ende eines Prozesspfades erreicht wurde.

Bereits bei diesen einfachen Ereignissen müssen wir eine weitere Unterscheidung treffen:

- Startereignisse sind stets eintretende Ereignisse (engl.: catching events). Das bedeutet, es ist etwas passiert, zwar unabhängig vom Prozess, aber der Prozess muss darauf warten bzw. reagieren.
- Zwischenereignisse können eintreten oder durch den Prozess selbst hervorgerufen bzw. ausgelöst werden (engl.: throwing events). Das Blanko-Zwischenereignis kennzeichnet einen Status, der durch den Prozessfortschritt erreicht wird, deshalb ist es stets ein ausgelöstes Ereignis. Später lernen wir andere Arten von Zwischenereignissen kennen, die wir als eintretende Ereignisse klassifizieren müssen.
- Endereignisse finden zu einem Zeitpunkt statt, an dem der Prozess nicht mehr auf sie reagieren kann. Folgerichtig können sie nur durch den Prozess ausgelöst werden und nicht unabhängig davon eintreten.

Unser BPMN-Knigge

Ereignisse beziehen sich auf etwas, was bereits passiert ist (unabhängig vom Prozess, wenn sie eingetreten sind, und dank des Prozesses, wenn sie ausgelöst wurden). Deshalb verwenden wir das [Objekt] und passivieren das [Verb], schreiben also beispielsweise „Hunger festgestellt“. Die BPMN schreibt nicht zwingend vor, dass Sie für einen Prozess ein Start- und ein Endereignis modellieren, Sie können diese auch weglassen. **Wenn** Sie aber ein Starterereignis modellieren, muss auch an jedem Pfadende ein Endereignis modelliert werden, und umgekehrt. Wir modellieren prinzipiell immer mit Start- und Endereignissen. Aus zwei Gründen: Erstens kann man somit explizit festhalten, wodurch ein Prozess ausgelöst wurde. Zweitens kann man den jeweiligen Endzustand beschreiben, der sich bei unterschiedlichen Pfadenden ergibt. Nur bei Teilprozessen verzichten wir mitunter auf diese Möglichkeit, aber dazu kommen wir später.

Sequenzflüsse

Der Sequenzfluss beschreibt die zeitlich-logische Reihenfolge, in der die Flusselemente (also Aufgaben, Ereignisse und die später beschriebenen Gateways) zueinander stehen.

Ein Sequenzfluss ist auch der Prozesspfad, über den unser Token wandert. Durch das Starterereignis wird es gemeinsam mit der Prozessinstanz „geboren“. Über den Sequenzfluss gelangt es über die Aufgaben und das Zwischenereignis zum Endereignis, wo es „konsumiert“ wird und verschwindet, was auch zum „Tod“ unserer Prozessinstanz führt.

Unser BPMN-Knigge

Wenn Sie möchten, können Sie BPDs auch vertikal zeichnen, anstatt wie in unseren Beispielen horizontal. Das ist zwar nicht üblich, aber auch nicht verboten. Wir zeichnen BPDs stets horizontal von links nach rechts.

2.3 Prozesspfade mit Gateways gestalten

2.3.1 Datenbasiertes exklusives Gateway

Die wenigsten Prozesse laufen immer gleich ab. Viel häufiger kommt es vor, dass die durchlaufenen Prozesspfade in den unterschiedlichen Prozessinstanzen variieren, weil bestimmte Dinge eben nur unter bestimmten Umständen zu erledigen sind.

In unserem einfachen Beispiel (Abbildung 2.3 auf der nächsten Seite) wollen wir uns mit der Kochkunst etwas näher beschäftigen. Getrieben vom Hunger, überlegen wir uns, was es heute geben soll. Da wir nur drei Rezepte kennen, suchen wir

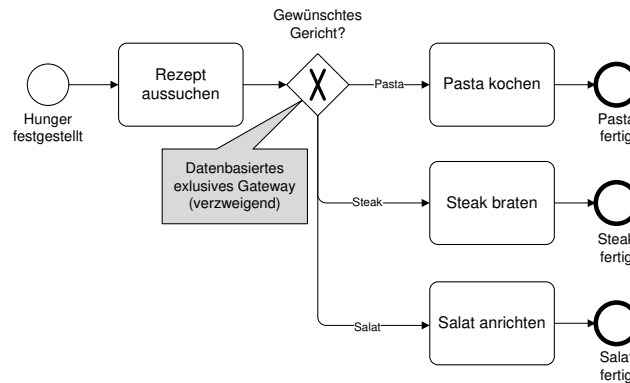


Abbildung 2.3: Das XOR-Gateway

uns eines davon aus. Je nachdem, für welches wir uns entscheiden, werden wir **entweder** Pasta kochen, ein Steak braten, **oder** einen Salat anrichten. Diese drei Möglichkeiten schließen sich gegenseitig aus – wir werden niemals mehr als eines dieser drei Gerichte zubereiten (ja, Sie denken bereits an den Salat als Beilage, aber gedulden Sie sich!). Der Punkt, an dem wir entscheiden, was als Nächstes zu tun ist, nennt sich Gateway. Da wir diese Entscheidung aufgrund verfügbarer Daten treffen (das ausgesuchte Rezept) und nur einer der ausgehenden Pfade durchlaufen wird, ist es ein datenbasiertes exklusives Gateway. In der Kurzfassung sprechen wir auch einfach vom XOR-Gateway (XOR = Exclusive OR).

Beachten Sie: Ein Gateway ist keine Aufgabe! Es basiert auf einer ganz einfachen Tatsache. Diese Tatsache herauszufinden, ist eine Aufgabe, die vor diesem Gateway erledigt werden muss. Dieses scheinbar banale Prinzip sollten Sie nicht vergessen, sondern bei der Modellierung stets berücksichtigen. Es wird uns zu einem späteren Zeitpunkt wieder begegnen, wenn wir uns mit dem Business Rules Management beschäftigen (siehe Abschnitt 4.5.4 auf Seite 178).

Leider ist die BPMN in Bezug auf XOR-Gateways etwas verwirrend: Es existieren hierfür nämlich zwei Symbole, die in ihrer Bedeutung identisch sind (siehe Abbildung 2.4). Wir verwenden stets die Version mit dem enthaltenen „X“, weil wir das für eindeutiger halten. Welche Sie verwenden, hängt von Ihrem Geschmack und Ihrem BPMN-Tool ab.

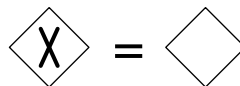


Abbildung 2.4: Beide Symbole bedeuten dasselbe

Unser BPMN-Knigge

Über dem Gateway haben wir die entscheidungsrelevante Frage platziert. Die möglichen Antworten auf diese Frage haben wir an die ausgehenden Pfade geschrieben. Lediglich letzteres wird in der Spezifikation genauso dargestellt, die Platzierung der entscheidungsrelevanten Frage über dem Gateway ist hingegen unsere eigene Konvention. In unseren Projekten hat sich diese Konvention aber bewährt, und wir arbeiten mit XOR-Gateways grundsätzlich immer nach dem folgenden Schema:

1. Schritt: Aufgabe modellieren, die die Entscheidungsgrundlage für das XOR-Gateway liefert.
2. Schritt: Dahinter das XOR-Gateway modellieren mit einer Frage, deren mögliche Antworten sich gegenseitig ausschließen.
3. Schritt: Pro mögliche Antwort einen ausgehenden Pfad (Sequenzfluss) modellieren, der mit der Antwort beschriftet wird.

Ein XOR-Gateway kann beliebig viele ausgehende Pfade haben. Dass wir in diesem Beispiel den ersten ausgehenden Pfad an der rechten Ecke und die übrigen an der unteren Ecke angedockt haben, hat keine weitere Bedeutung – es entspricht einfach unserer Stil-Konvention.

Dass dieser Prozess drei Endereignisse besitzt, ist übrigens nicht ungewöhnlich. Er kann eben drei verschiedene End-Status erzeugen, und es kann gerade bei komplexeren Diagrammen sehr hilfreich sein, dies auf einen Blick zu erkennen. Wir werden später noch weitere Gründe kennenlernen, warum die Arbeit mit unterschiedlichen Endereignissen sinnvoll ist. Wie man sieht, ist die BPMN also keine „blockorientierte“ Prozessnotation. Das bedeutet, es besteht keinerlei Zwang, einen aufgegabelten Prozesspfad zu einem späteren Zeitpunkt zusammenzuführen. Sie können das tun, müssen aber nicht.

Natürlich kann es aus semantischen Gründen sinnvoll sein, die drei Pfade wieder zusammenzuführen. Wenn beispielsweise nach der Zubereitung die Mahlzeit verzehrt wird, passiert das ja in jedem Fall, ganz unabhängig vom ausgesuchten Rezept. Auch für eine solche Zusammenführung können wir das XOR-Gateway verwenden. Es sorgt dafür, dass jedes Token, das von einem der drei eingehenden Pfade kommt, auf den einen einzigen ausgehenden Pfad geleitet wird (siehe Abbildung 2.5 auf der nächsten Seite).

Die doppelte Verwendungsmöglichkeit des XOR-Gateways (verzweigend oder zusammenführend, oder auch kurz einfach „XOR-Split“ und „XOR-Join“) ist für Anfänger manchmal etwas verwirrend. Sie dürfen sogar ein XOR-Gateway modellieren, das in einem Rutsch sowohl zusammenführt als auch verzweigt (siehe Abbildung 2.6 auf der nächsten Seite)! Ob Sie diese Möglichkeit für eine kompakte Diagrammform bevorzugen, müssen Sie selbst entscheiden. Wir verzich-

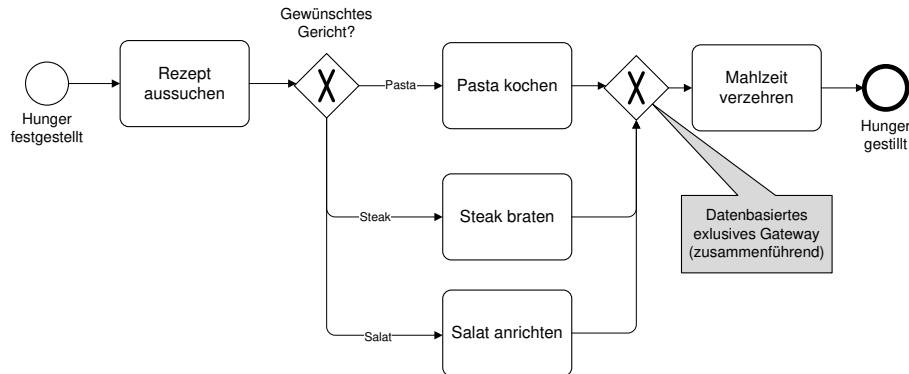


Abbildung 2.5: XOR-Gateways können auch zusammenführen

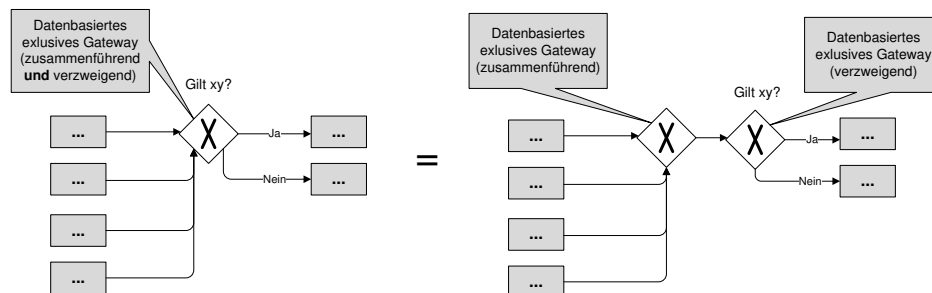


Abbildung 2.6: Eine kombinierte Zusammenführung/Verzweigung kann man auf zwei Arten darstellen.

ten meistens darauf und zeichnen lieber zwei XOR-Gateways hintereinander, um Fehlinterpretationen vorzubeugen.

2.3.2 Paralleles Gateway

Was machen wir, wenn wir den Salat als Beilage wünschen? Gehen wir mal vom einfachen Fall aus, dass der Salat auf jeden Fall gewünscht ist, dann könnte man das natürlich wie in Abbildung 2.7 auf der nächsten Seite gezeigt modellieren.

Bei der Gelegenheit haben wir gleich mal ein weiteres Symbol eingeführt, die (Text-)Anmerkung. Das ist ein Artefakt, das Sie bekanntlich via Assoziation an ein beliebiges Flussobjekt (hier: Aufgaben) andocken können. In die Anmerkung können Sie hineinschreiben, was immer Sie wollen. In diesem Beispiel haben wir einmal eingetragen, wie viel Zeit wir für die Bearbeitung der einzelnen Aufgaben im Durchschnitt benötigen. Die Summe dieser Bearbeitungszeiten ergibt die Durchlaufzeit des Prozesses, in diesem Fall liegt sie also bei 48 Minuten, wenn wir uns für die Pasta entscheiden, und bei 43 Minuten, wenn wir uns mal eben

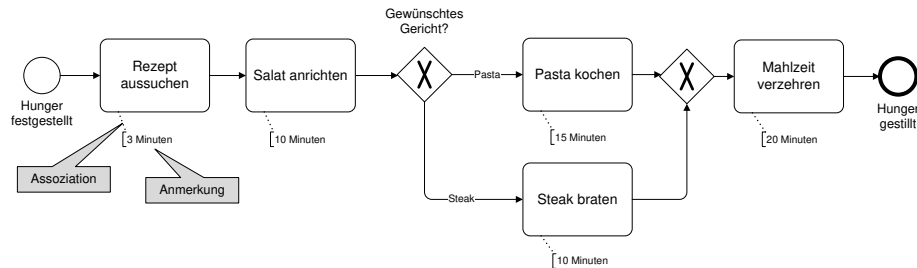


Abbildung 2.7: Zubereitung von Salat und Hauptgericht

schnell ein Steak braten. Gratulation, Sie haben soeben Ihre erste Erfahrung in der kennzahlenbasierten Prozessanalyse gemacht.

Das bedeutet also, es dauert 28 bzw. 23 Minuten, bis wir unsere Mahlzeit verzehren können. Das kann unerträglich lang sein, wenn man großen Hunger hat. Was tun? Sinnvollerweise sollten wir mit der Zubereitung von Pasta bzw. Steak nicht erst beginnen, wenn der Salat fertig ist, sondern beides gleichzeitig erledigen, das lässt sich ja durchaus parallelisieren. Das Symbol dafür ist das parallele Gateway, das man auch kurz als „AND-Gateway“ bezeichnen kann, zu sehen in Abbildung 2.8.

Die Parallelisierung bedeutet nicht, dass die Aufgaben zwangsläufig gleichzeitig ausgeführt werden **müssen** – aber im Gegensatz zum Beispiel in Abbildung 2.7 ist es auch nicht zwingend erforderlich, zuerst den Salat zuzubereiten und danach erst die übrigen Aufgaben anzugehen. Für die Berechnung unserer Durchlaufzeit bedeutet das natürlich eine Verkürzung um 10 Minuten. Wie Sie sich vorstellen

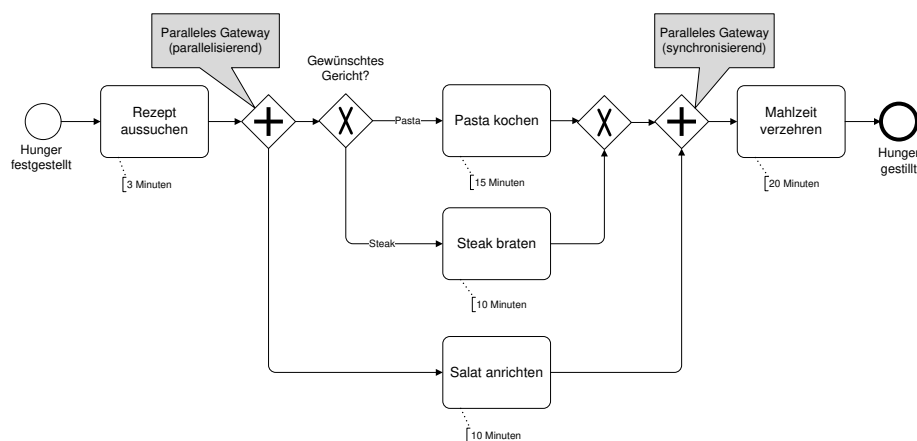


Abbildung 2.8: Der Salat wird gleichzeitig mit dem Hauptgericht zubereitet

können, ist es ein Klassiker der Prozessoptimierung, nach Möglichkeit alle Aufgaben zu parallelisieren, die nicht zwingend aufeinander aufbauen.

Im gezeigten Beispiel wird der Prozess nicht nur parallelisiert (kurz: „AND-Split“), zu einem späteren Zeitpunkt werden die Pfade auch wieder synchronisiert („AND-Join“). Der Grund ist nachvollziehbar: Erst wenn sowohl das Hauptgericht als auch die Beilage zubereitet sind, kann mit dem Verzehr der Mahlzeit begonnen werden.

Wie würde in einer Instanz dieses Prozesses das Token-Konzept wirken? Das Token wird wieder beim Startereignis „geboren“, es durchläuft die Aufgabe „Rezept aussuchen“ und wandert dann in den AND-Split. Dort werden aus dem einen Token so viele Token, wie Pfade aus dem Gateway laufen, in diesem Fall also zwei. Das erste Token wandert nun weiter in den XOR-Split, wo es je nach gewünschtem Rezept auf einen der ausgehenden Pfade geschickt wird. Nehmen wir mal an, es soll Pasta gekocht werden, so läuft das Token in diese Aufgabe und verharrt dort 15 Minuten. Gleichzeitig ist das zweite Token nach unten in die Aufgabe „Salat anrichten“ gewandert, wo es nur 10 Minuten bleibt. Nach diesen 10 Minuten wandert es weiter bis zum synchronisierenden AND-Join. Die Anzahl der eingehenden Pfade bestimmt, auf wie viele zusammengehörende Token das Gateway wartet. In diesem Fall wartet es also auf zwei Token, die zur selben Prozessinstanz gehören müssen. Da in unserem Szenario das zweite Token schon nach 10 Minuten beim AND-Join eintrifft, während das erste insgesamt 15 Minuten in der Aufgabe „Pasta kochen“ verharrt, wartet der AND-Join also noch 5 Minuten, bis das erste Token ebenfalls ankommt. Erst dann werden die beiden vom AND-Join zu einem einzigen Token verschmolzen, das auf den ausgehenden Pfad geschickt wird.

Klingt das jetzt sehr abstrakt oder technisch? Ist es nicht. Das Verhalten des AND-Join ist identisch mit Ihrem eigenen Verhalten: der Salat ist fertig, die Pasta noch nicht. Also warten Sie. Wenn auch die Pasta endlich fertig ist, kann gegessen werden. Dasselbe Prinzip.

Warum also das scheinbar komplizierte Token-Konzept? Denken Sie beispielsweise an die 90 Mio. Prozessinstanzen, die jährlich bei der Schufa erzeugt werden. Diese werden natürlich nicht streng nacheinander, sondern überlappend ausgeführt. Wenn diese komplexen Prozesse mit ihren diversen Parallelisierungen, Verzweigungen, Zusammenführungen und Synchronisationen fehlerfrei definiert und täglich abgewickelt werden sollen, ist der Token-Ansatz in der Konzeption und Umsetzung der Prozesse nicht nur extrem hilfreich, sondern auch notwendig. Jetzt sollte auch klar geworden sein, dass eine Prozessinstanz nicht dasselbe wie ein Token ist: Im Rahmen einer Prozessinstanz können durchaus mehrere Token laufen.

Wir sollten Ihr Token-Verständnis noch mit zwei Testfragen erproben.

Frage: In Abbildung 2.9 auf der nächsten Seite ist derselbe Prozess dargestellt, allerdings wurde aus Platzgründen auf den AND-Join verzichtet, und der Pfad aus

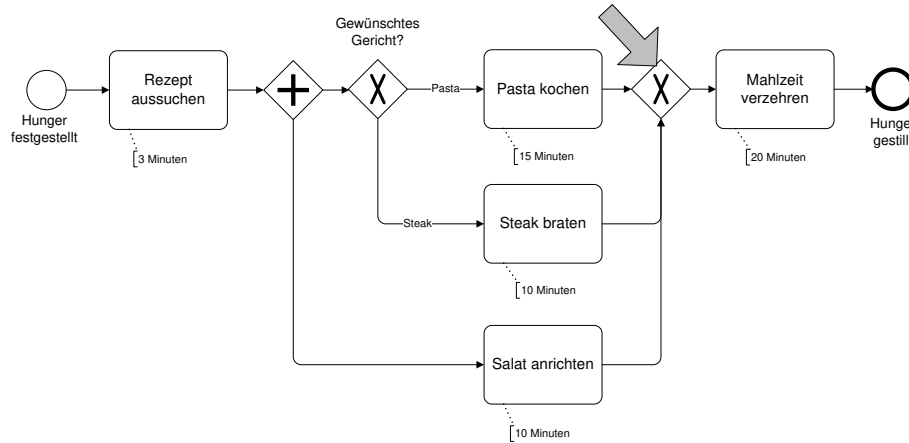


Abbildung 2.9: Was passiert in diesem Prozess?

der Aufgabe „Salat anrichten“ wird direkt in den XOR-Join geleitet. Was passiert, wenn wir den Prozess instanziiert (wir entscheiden uns für die Pasta)?

Antwort: Mit der Prozessinstanz wird das Token erzeugt, das wie gehabt beim AND-Split geklont wird. Sobald der Salat angerichtet ist, wird das Token über den XOR-Join geleitet und die Aufgabe „Mahlzeit verzehren“ ausgeführt. 5 Minuten später ist auch die Aufgabe „Pasta kochen“ abgeschlossen. Das Token wird ebenfalls über den XOR-Join geleitet, und die Aufgabe „Mahlzeit verzehren“ wird erneut ausgeführt! Nicht gerade das Verhalten, das wir uns gewünscht haben.

Frage: In Abbildung 2.10 ist ein Prozess dargestellt, der nur aus zwei Aufgaben besteht. Der Prozess wird instanziiert. Wie lange „lebt“ nun die Prozessinstanz?

Antwort: Sie „lebt“ 45 Tage, was dementsprechend die Durchlaufzeit des Prozesses ist. Obwohl das erste im AND-Split erzeugte Token die Aufgabe 1 bereits nach 30 Tagen durchlaufen hat und danach vom oberen Endereignis „konsumiert“ wird, hält sich das zweite Token noch weitere 15 Tage in Aufgabe 2 auf. Danach

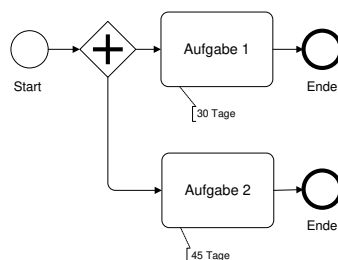


Abbildung 2.10: Wie lange „lebt“ die Prozessinstanz?

läuft es erst zum unteren Endereignis und wird konsumiert, was das Ende der Prozessinstanz bedeutet.

Merke: Solange innerhalb des Prozesses noch ein Token „lebt“, „lebt“ auch die Prozessinstanz! Erst wenn alle erzeugten Token wieder konsumiert wurden, ist die Instanz beendet.

2.3.3 Datenbasiertes inklusives Gateway

Wir wollen unseren Prozess noch etwas flexibler gestalten: Wenn wir Hunger bekommen, wollen wir

- nur einen Salat oder
- einen Salat und „etwas Ordentliches“ wie Pasta oder Steak oder
- nur „etwas Ordentliches“

essen. Mit den Symbolen, die Sie bisher kennengelernt haben, könnten Sie das wie in Abbildung 2.11 gezeigt modellieren.

Wenn wir eine kompaktere Darstellung wünschen, können wir das datenbasierte inklusive Gateway verwenden, kurz OR-Gateway genannt (siehe Abbildung 2.12 auf der nächsten Seite). Mit dem OR-Gateway können wir eine Und-Oder-Situation beschreiben, bei der wir entweder einen, mehrere oder auch alle ausgehenden Pfade gleichzeitig durchlaufen können. Insofern können wir es gut gebrauchen, um eine besondere Komplexität der Diagramme zu vermeiden. Die kombinierte Wirkung des OR-Gateways nutzen wir auch, wenn die Pfade wieder zusammen laufen: Je nachdem, ob wir lediglich einen Salat **oder** etwas Ordentliches oder aber einen Salat **und** etwas Ordentliches essen wollen, müssen wir

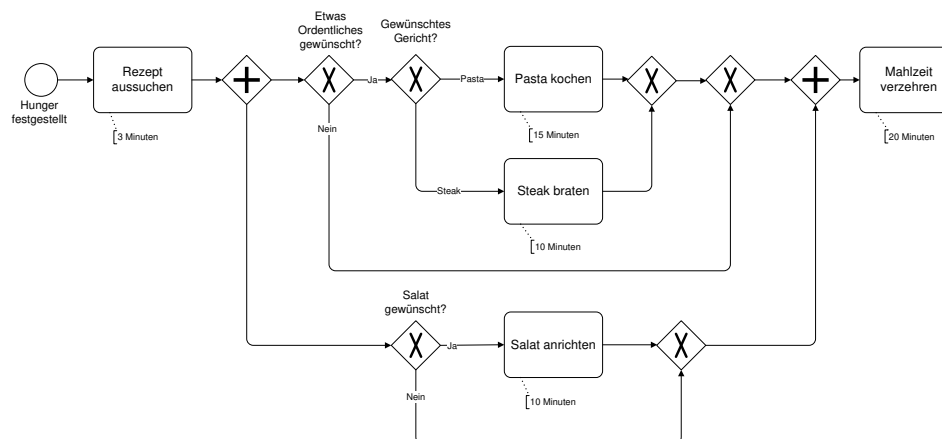


Abbildung 2.11: Unterschiedliche Optionen bei der Zusammenstellung unserer Mahlzeit

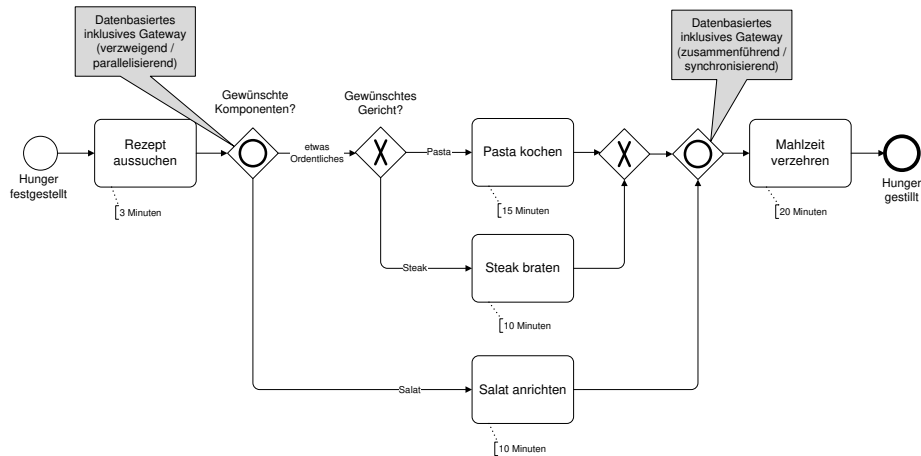


Abbildung 2.12: Das OR-Gateway ermöglicht eine kompakte Darstellung komplexer Pfadvarianten.

vor dem Verzehrer der Mahlzeit entweder nur auf ein eingehendes Token warten (Zusammenführung) oder aber auf beide (Synchronisation). Hinweis: Einen Unterschied zum Modell in Abbildung 2.11 auf der vorherigen Seite gibt es in diesem Prozess allerdings. In der Fassung ohne OR-Gateway konnten wir uns auch dazu entschließen, überhaupt nichts zuzubereiten (also weder Salat noch etwas Ordentliches). Trotzdem hätten wir nach dieser Entscheidung die Mahlzeit verzehrt, was natürlich unsinnig ist. In der Variante mit dem OR-Gateway ist dieser Fall ausgeschlossen – wir müssen uns wenigstens für einen Salat und/oder etwas Ordentliches entscheiden, ansonsten bleibt das Token im Gateway stecken.

Wie Sie sich vorstellen können, ist der praktische Umgang mit dem OR-Gateway nicht immer ganz einfach. In diesem simplen Beispiel ist es leicht nachvollziehbar, unter welchen Umständen am OR-Join auf ein weiteres Token gewartet werden muss, bevor es weitergehen kann. In komplexen Diagrammen jedoch, die sich vielleicht über zahlreiche Seiten hinweg erstrecken, kann es ziemlich schwierig werden, die Regeln der Synchronisation nachzuvollziehen. Die Lösung besteht auch nicht darin, sich einfach zu merken, welche Bedingungen beim OR-Split galten. Schauen Sie sich einmal Abbildung 2.13 auf der nächsten Seite an. Je nachdem, ob nach dem OR-Split einer oder mehrere Pfade durchlaufen werden, muss der OR-Join synchronisieren, oder nicht.

Angenommenes Szenario: Nach 30 Tagen kommt das erste Token am OR-Join an. Weil beim vorherigen OR-Split auch Antwort 2 galt, ist ein weiteres Token unterwegs, das sich noch für 15 Minuten in der Aufgabe 2 befindet. Diese ist abgeschlossen. Jetzt kann es aber passieren, dass am XOR-Split eine Entscheidung getroffen wird, die dazu führt, dass das Token über den Pfad „Antwort 1“ geleitet und vom Endereignis konsumiert wird. Was passiert jetzt mit dem ersten Token

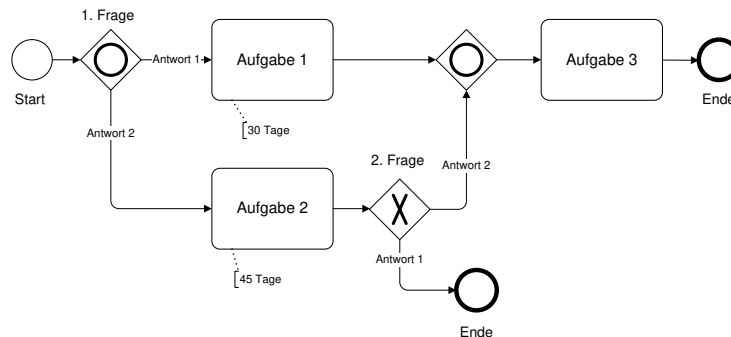


Abbildung 2.13: Wie lange muss das zweite OR-Gateway warten?

am synchronisierenden OR-Join? **Das OR-Gateway muss registrieren, dass das zweite Token verschwunden ist, und das erste Token weiterleiten.**

Und das kann für drei Szenarien problematisch sein:

- Wenn Sie in Ihrem Prozesshandbuch auf Seite 10 einen OR-Join vorfinden und die letzten 9 Seiten sichten müssen, um zu verstehen, unter welchen Umständen dort wie lange gewartet wird.
- Wenn Sie einen solchen Prozess organisatorisch umsetzen – wer sagt der für Aufgabe 3 verantwortlichen Person Bescheid, dass sie den Prozess wider Erwarten bereits fortsetzen kann?
- Wenn der Prozess in einer Process Engine abgewickelt wird, muss die Software das Synchronisationsverhalten steuern – eine derartige Prüfung umzusetzen, ist aufwendig und fehlerträchtig und in bestimmten Fällen sogar unmöglich.

Es sprechen also einige Argumente dafür, das OR-Gateway nicht übermäßig, sondern mit Bedacht einzusetzen.

Frage: Könnten wir den Prozess nicht auch so modellieren, wie in Abbildung 2.14 auf der nächsten Seite dargestellt?

Antwort: Sicher, das würde das Modell noch kompakter machen. Die Bedeutung wäre jedoch etwas anders. Laut diesem Prozessmodell gäbe es folgende Varianten:

- Wir essen nur Pasta.
- Wir essen nur Steak.
- Wir essen nur Salat.
- Wir essen Pasta und Salat.
- Wir essen Steak und Salat.
- Wir essen Pasta und Steak.

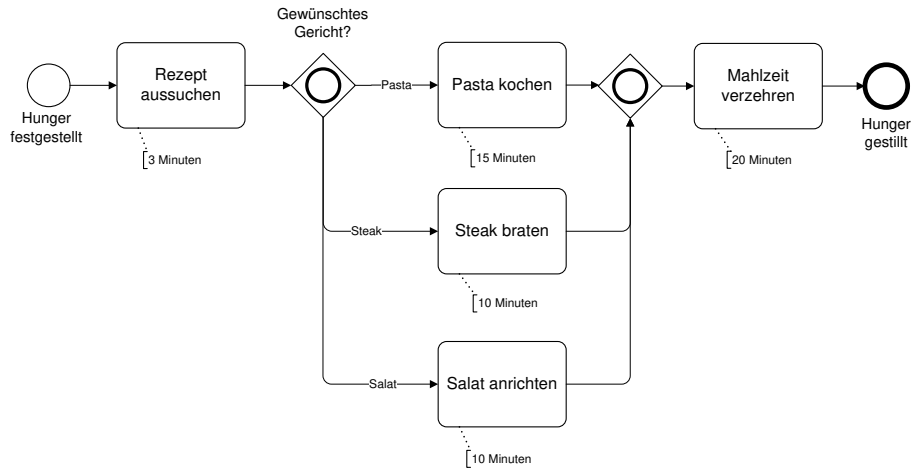


Abbildung 2.14: Eine wunderbar (?) kompakte Version

■ Wir essen Pasta, Steak und Salat.

Die letzten beiden Varianten entsprechen also nicht dem, was wir eigentlich definieren wollten.

2.3.4 Standardfluss und Steckenbleiben

Wir sollten uns einen weiteren Aspekt bei der Arbeit mit XOR- bzw. OR-Gateways anschauen. Der Einfachheit halber lassen wir im nächsten Beispiel das Thema Salat außer Acht und konzentrieren uns auf ordentliche Mahlzeiten.

Was passiert, wenn wir weder Pasta noch Steak essen wollen? In den bisherigen Modellen hätte diese Situation dazu geführt, dass unser Token niemals über den XOR-Split („Gewünschtes Gericht“) hinausgekommen wäre. Es wäre ewig stecken geblieben, was natürlich nicht sein kann. Die BPMN-Spezifikation ist leider nicht sehr präzise, wie in einem solchen Fall zu verfahren ist; sie konstatiert:

BPMN legt nicht fest, was passiert, wenn kein ausgehender Pfad durchlaufen werden kann. Allerdings spezifiziert die BPMN, dass es keine impliziten Flüsse geben darf und dass alle normalen Flüsse durch Sequenzflüsse dargestellt werden. Das würde bedeuten, dass ein Prozesmodell mit einem Gateway, bei dem zur Laufzeit des Prozesses möglicherweise kein ausgehender Pfad durchlaufen werden kann, ungültig ist.

Mit „impliziten Flüssen“ sind Prozesspfade gemeint, die nicht mit Sequenzflüssen ausmodelliert werden. Laut Spezifikation gibt es Notationen, bei denen ein Token, das an einem XOR-Split „hängt“, weil es keinen ausgehenden Pfad nehmen kann,

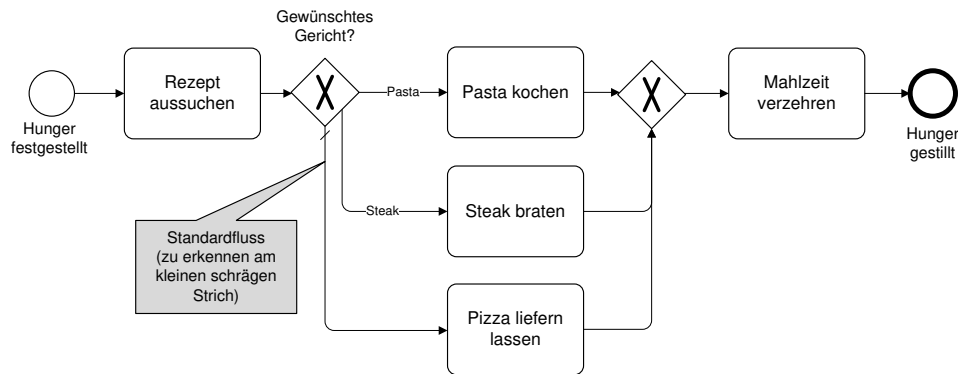


Abbildung 2.15: Der Standardfluss

einfach zum Ende des Prozesses springt. Eine derart schwammige Semantik ist in BPMN natürlich (zum Glück) nicht erlaubt. Im Entwurf zu BPMN 2.0 ist der Passus mit dem „ungültigen Diagramm“ verschwunden, dafür wird definiert, was bei einer solchen Situation im akuten Fall passiert:

Falls keine Bedingung zutrifft (kein ausgehender Pfad genommen werden kann, Anm.d.Ü.) und kein Standardfluss spezifiziert wurde, wird eine Exception geworfen.

Bitte werden Sie jetzt nicht wütend, weil von „Exception werfen“ die Rede ist! Wir werden uns zu einem späteren Zeitpunkt mit diesem Thema beschäftigen und zeigen, dass es absolut nicht nur die IT betrifft.

Der Standardfluss ist auch in der aktuellen BPMN-Fassung bereits verfügbar, und er „beschützt“ uns vor der Gefahr des Steckenbleibens. In Abbildung 2.15 haben wir ihn verwendet, erkennbar am kleinen schrägen Strich. Das Prinzip des Standardflusses ist einfach: Zuerst werden alle anderen ausgehenden Pfade betrachtet. Nur wenn keiner dieser Pfade infrage kommt, wird der Standardfluss durchlaufen.

Bitte machen Sie aufgrund der Bezeichnung nicht den Fehler, den „Standardfluss“ mit dem „üblichen“ Fluss zu verwechseln. Das Symbol trifft keinerlei Aussage darüber, ob dieser Prozesspfad in den meisten Prozessinstanzen durchlaufen wird, das ist eine ganz andere Frage.

Sollte man also besser gleich auf diesen Ansatz verzichten und grundsätzlich immer mit Gateways arbeiten? Nein, das muss auch nicht sein. Gerade wenn es um einfache Prozessmodelle geht, können die Gateways tatsächlich mehr Verwirrung als Nutzen stiften. Einfache Schleifen zum Beispiel lassen sich ohne XOR-Join besser darstellen, da dieser den unbedarften Leser häufig verwirrt. Außerdem dürfen wir in BPMN auch mehrere Sequenzflüsse aus Startereignissen heraus- bzw. in Endereignisse hineinlaufen lassen, was in Summe zu wesentlich kompakteren Diagrammen führen kann. In Abbildung 2.23 auf der vorherigen Seite ist derselbe Prozess einmal mit und einmal ohne Gateways modelliert, um dies zu veranschaulichen. Streng genommen sind die beiden Modelle jedoch nicht identisch: Im oberen schließt das XOR-Gateway syntaktisch aus, dass mehrere Pfade durchlaufen werden können. Es setzt also voraus, dass Bedingung 1 und Bedingung 2 niemals gleichzeitig auftreten. Im unteren Prozessmodell ist das nicht der Fall, hier könnten auch beide Bedingungen eintreten.

2.5 Lanes

Bislang haben wir darüber gesprochen, **was** in unseren Prozessen zu tun ist. Völlig ungeklärt blieb bisher, **wer** für die Erledigung der einzelnen Aufgaben zuständig ist. Diese Frage kann in BPMN mit Hilfe von Lanes beantwortet werden.

In Abbildung 2.24 auf der nächsten Seite ist zu sehen, dass die Aufgaben unseres Beispielprozesses bestimmten Personen zugeordnet wurden. Aus dieser Zuordnung können wir die folgende Prozessbeschreibung ableiten: Wenn Christian Hunger hat, sucht er sich ein bestimmtes Rezept aus. Je nachdem, aus welchen Komponenten seine Mahlzeit bestehen soll, kann er sich entweder selbst darum kümmern (Pasta kochen), oder er spannt seine Mitbewohner ein: Falko muss dann das Steak braten, bzw. Robert ist für den Salat zuständig. Am Ende kann Christian die Mahlzeit verzehren. Die drei Lanes (Christian, Falko, Robert) sind in einem Pool mit der Bezeichnung „Wohngemeinschaft“ zusammengefasst. Pools besprechen wir ausführlich in Abschnitt 2.9 auf Seite 95.

In diesem Beispiel wurden die Lanes zwar mit Personen gleichgesetzt. Diese Semantik ist aber nicht durch die BPMN vorgegeben – Sie können die Lanes bezeichnen, wie Sie möchten. Lanes werden in der Praxis häufig auch für folgende Zuordnungen verwendet:

- Stellen der Primärorganisation, z.B. „Sachbearbeiter Buchhaltung“
- Rollen der Sekundärorganisation, z.B. „Datenschutzbeauftragter“
- Allgemeine Rollen, z.B. „Kunde“
- Abteilungen, z.B. „Vertrieb“
- IT-Anwendungen, z.B. „CRM-System“

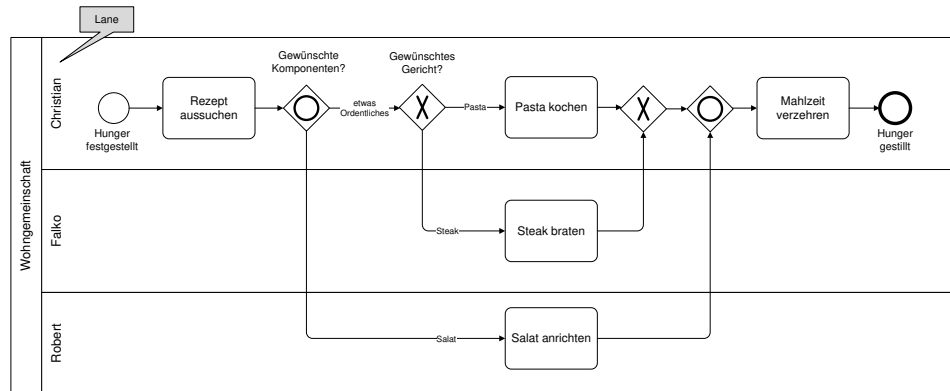


Abbildung 2.24: Mit Lanes können Zuständigkeiten dargestellt werden.

BPMN-Tooling

Manche Tools erlauben es Ihnen, die Elemente in Ihrem BPD unterschiedlichen Kategorien oder Sichten zuzuordnen (z.B. durchführende Stellen, verantwortliche Stellen, unterstützende IT-Anwendungen etc.) und den Prozess in der jeweiligen Sicht anzuzeigen. Dadurch werden unterschiedliche Lanes eingeblendet und die Elemente entsprechend angeordnet.

Der Begriff „Lane“ besitzt in der Welt der Prozessmodellierung übrigens eine gewisse Historie: Als „Swimlane-Darstellung“ werden generell verschiedene Prozessnotationen bezeichnet, die eine Aufgabenzuordnung nach dem dargestellten Prinzip vornehmen. Dem Begriff liegt die Analogie zu einem Schwimmbecken zugrunde, in dem die einzelnen Schwimmer nur in den ihnen zugewiesenen Bahnen schwimmen.

In BPMN können Lanes auch verschachtelt sein, um eine Verfeinerung der Zuständigkeiten darzustellen (siehe Abbildung 2.25 auf der nächsten Seite).

Unser BPMN-Knigge

Auch in Bezug auf die vertikale Reihenfolge von Aufgaben macht Ihnen die BPMN keine Vorschriften: In Abbildung 2.25 auf der nächsten Seite beginnt der Prozess links oben und endet rechts unten, was unserer Konvention entspricht. Sie können ihn aber genauso gut von links unten nach rechts oben modellieren. Wie immer ist es entscheidend, dass Sie sich für einen Stil entscheiden und diesen konsequent anwenden, damit die Diagramme einheitlich aufgebaut und somit einfacher lesbar sind.

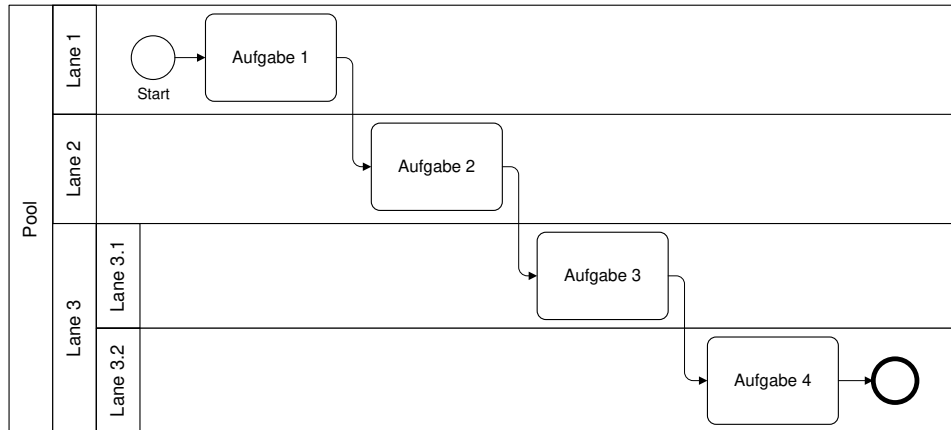


Abbildung 2.25: Verschachtelung von Lanes

Der Umgang mit Lanes ist in der Praxis oft verwickelter, als man zunächst annehmen darf. In unserem kleinen Prozess beispielsweise gehen wir von einer klaren Aufteilung der Aufgaben aus. Aber was machen wir, wenn auch Falko und Robert essen wollen? Syntaktisch falsch wäre eine Darstellung wie in Abbildung 2.26 – das dürfen Sie nicht machen. Ein Flussobjekt (Aktivität, Ereignis, Gateway) darf immer nur innerhalb einer Lane positioniert sein.

Die Lösung für dieses Szenario besteht darin, die Aufgabe mehrmals anzulegen und den jeweiligen Personen zuzuordnen (Abbildung 2.27 auf der nächsten Seite). Das ergibt auch inhaltlich einen Sinn, weil die Aufgabe ja nun tatsächlich dreimal ausgeführt wird. Diese Darstellung kann allerdings auch zu Missverständnissen führen: Es ist nicht direkt ersichtlich, dass alle drei gemeinsam essen. In diesem

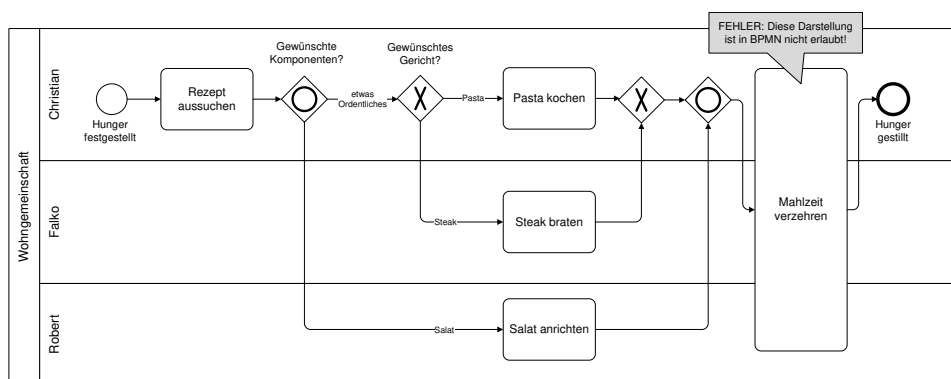


Abbildung 2.26: Falscher Umgang mit Lanes

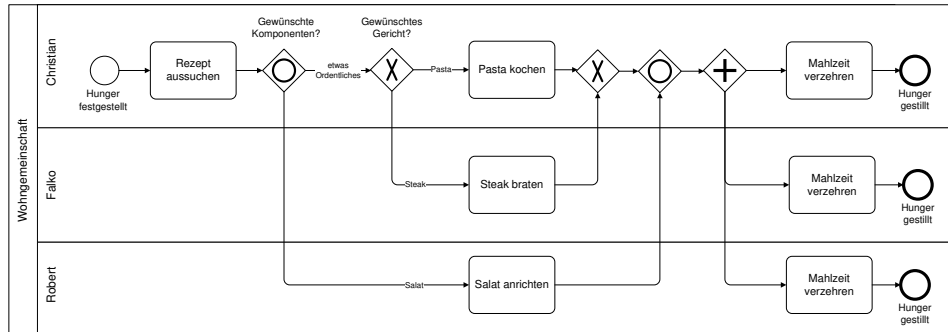


Abbildung 2.27: Richtiger Umgang mit Lanes

Fall ist das vielleicht nicht kritisch. Aber wenn eine tatsächliche Zusammenarbeit stattfinden soll, beispielsweise die Aufgabe „Gutachten erstellen“, ist nicht mehr klar, ob jeder ein eigenes Dokument erstellt oder alle ein gemeinsames. In diesem Fall könnte man sich mit einem Gruppierungsrahmen behelfen, den wir in Abschnitt 2.10.2 auf Seite 107 vorstellen.

Hinweis: In unseren BPDs sind die Beschriftungen für die Lanes nicht von der Lane selbst getrennt. Dies entspricht dem Entwurf zur BPMN 2.0, wo eine solche Trennung explizit verboten wurde. Bis zur BPMN 1.2 war sie hingegen erlaubt, weshalb es durchaus sein kann, dass Sie in Ihrer BPMN-Praxis auch mal Diagramme wie in Abbildung 2.28 zu sehen bekommen bzw. mit Ihrem Tool nur solche Diagramme zeichnen können. Das Verbot der Trennlinie gehört zu den Neuerungen an BPMN 2.0, deren Nutzen wir nicht wirklich nachvollziehen können.

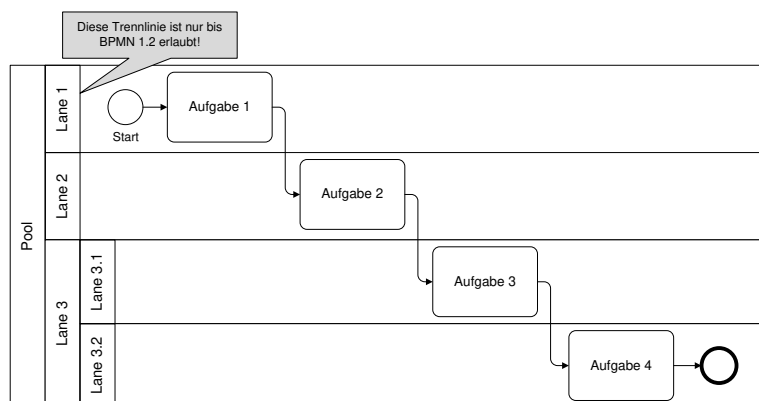


Abbildung 2.28: Die Trennlinien zwischen Lane-Header und Lane-Body sind nur bis BPMN 1.2 erlaubt.