Methoden zur adaptiven Anpassung von EPKs an individuelle Anforderungen vor der Abarbeitung

Jens Brüning
Peter Forbrig

Fakultät für Informatik und Elektrotechnik
Universität Rostock
Albert-Einstein-Str. 21
18059 Rostock

Jens.Bruening@uni-rostock.de
Peter.Forbrig@uni-rostock.de

Abstract: In diesem Artikel werden Methoden vorgestellt, wie Prozessmodelle an individuelle Anforderungen vor der Abarbeitung des Workflows angepasst werden können. Bei vielen Prozessen entscheidet sich dann, welche Teile von der zugehörigen Ereignisgesteuerten Prozesskette (EPK) ausgeführt werden und welche nicht. Beispielsweise entscheidet der Kunde zu diesem Zeitpunkt häufig, welche von den angebotenen Dienstleistungen er in Anspruch nehmen möchte. Außerdem könnten zusätzliche Anforderungen vom Kunden aufgestellt werden, die im bisherigen Prozessmodell noch nicht berücksichtigt wurden. Um Prozessmodelle zu erhalten, die den individuellen Anforderungen genügen, werden im Folgenden Methoden vorgeschlagen und an bestimmten Stellen die Sprachmittel der Ereignisgesteuerten Prozesskette erweitert, um Prozessdefinitonen zu Beginn des Prozesses adaptiver zu machen.

1 Einleitung

Workflowmodelle dienen dem Angestellten als Anleitung, welche Aktivitäten er in welcher Abfolge bei bestimmten Geschäftsprozessen leisten soll, damit ein geregelter Ablauf im Unternehmen garantiert ist. Sie spiegeln des Weiteren Erfahrungen wieder, wie bestimmte Geschäftsvorfälle mit der angegebenen Prozessdefinition zum Erfolg geführt werden. Die Modelle können in unterschiedlichen Sprachen vorliegen. So sind Ereignisgesteuerte Prozessketten (EPKs), UML Aktivitätsdiagramme, Business Process Modelling Notation (BPMN) und Petrinetze zurzeit sehr verbreitet, um Workflows auszudrücken. Im weiteren Verlauf dieses Artikels werden EPKs für Prozessdefinitionen verwendet und insbesondere im Kapitel 3 erweitert.

Konventionelle, nicht adaptive Workflowspezifikationen und deren Anwendungen in Workflow Management Systemen (WfMS) sind teilweise zu starr, so dass auf wechselnde Anforderungen nicht flexibel reagiert werden kann.

Die in diesem Artikel vorgestellten Vorschläge sollen helfen, wechselnde Anforderungen unmittelbar vor Beginn des Prozesses in die Prozessdefinition einfließen zu lassen. Es können dann Prozessteile aus dem Workflow anhand von neuen Operatoren ausgeschlossen oder hinzufügt werden. Der Idee, zum Anfang des Geschäftsprozesses Teile der Prozessdefinition auszulassen oder hinzuzufügen, liegt die Beobachtung zu Grunde, dass häufig kurz vor Beginn des Prozesses die Bedingungen mit dem Kunden geklärt werden und dieser dann angibt, welche Leistungen er nicht bzw. speziell noch zusätzlich erbracht haben möchte.

Der weitere Verlauf des Artikels ist wie folgt gegliedert: In Kapitel 2 werden die Grundlagen für die betrachteten EPKs kurz erläutert und erklärt, wie die dort vorhandenen Sprachmittel verwendet werden. Außerdem wird ein Fallbeispiel vorgestellt, das im Verlauf des Artikels zur Verdeutlichung der neuen Konzepte dienen soll. Im Kapitel 3 werden die neuen Konzepte eingeführt, mit denen man zur Instanziierungszeit Teile der Prozessdefinition nach den dort vorgestellten Regeln ausschließen oder hinzufügen und ggf. neu strukturieren kann. Kapitel 4 enthält schließlich eine Zusammenfassung und einen Ausblick, wie die neuen Konzepte umgesetzt werden können.

2 Grundlagen

Bei den Ereignisgesteuerten Prozessketten werden Workflows mit Ereignissen und Funktionen modelliert, die alternierend mit gerichteten Kanten verbunden werden. Der Funktionsbegriff wird in EPKs nach [KNS92] synonym zu Aufgaben verwendet. Des Weiteren gibt es Konnektoren, die den Kontrollfluss bzw. die Kanten aufspalten und zusammenführen können. Der AND-Konnektor kann parallele Prozessabläufe bzw. gegenseitig voneinander unabhängige Aufgaben modellieren. OR- und XOR-Konnektoren können alternativ auszuführende Aufgaben im Geschäftsprozess modellieren. Zusätzlich gibt es Hierarchisierungskonzepte durch Funktions-verfeinerungen, um die EPKs lesbarer zu machen. Genauere Einführungen in EPKs sind u.a. in [KNS92, NR02] zu finden.

Ein Beispiel für einen Geschäftsprozess ist in Abbildung 1 zu sehen. Es wurde ein mögliches Prozessmodell für eine Autowerkstatt erstellt, das in Abbildung 1a) angibt, wie mit einem ankommenden Auto zu verfahren ist. Die Funktion „Auto inspizieren und reparieren“ ist in Abbildung 1b) verfeinert, womit hier das Hierarchisierungskonzept angewendet wurde.

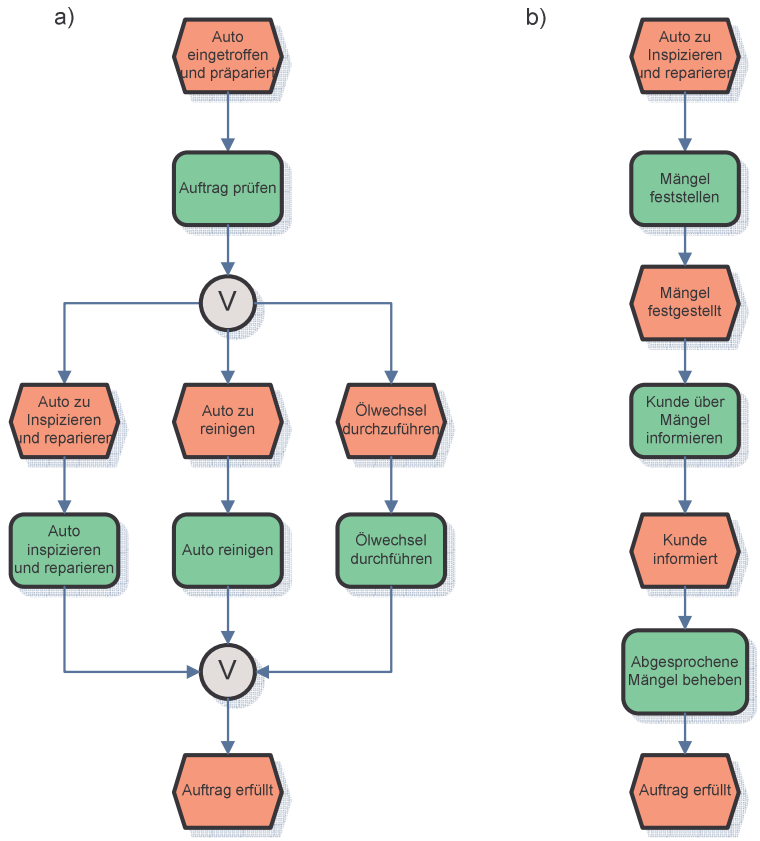

Abbildung 1: a) EPK Prozessdefinition zum Fallbeispiel: Auto inspizieren, reparieren, reinigen und Öl wechseln. b) Prozessverfeinerung für Prozess „Auto inspizieren und reparieren“

Der strukturierte Ablauf des Geschäftsprozesses ist mit den gerichteten Pfeilen modelliert, auf denen zur Ablaufzeit (Runtime) von den Anfangsereignissen ähnlich wie bei Petrinetzen Marken (bei EPKs auch Prozessordner genannt) über die Ereignisse, Funktionen und Kanten wandern. Das EPK-Prozessmodell an sich stellt hierbei noch keinen konkreten im Ablauf befindlichen Geschäftsprozess dar, sondern ist zunächst erst die Schablone dafür. Um einen konkreten Geschäftsprozess zu erhalten, muss dieser instanziiert werden. Erst dann befinden sich die Marken bzw. Prozessordner in der EPK, die jeweils kennzeichnen, welche Funktionen zur entsprechenden Zeit aktiv sind [NR02]. Das Instanziierungsprinzip von Prozessmodellen ist auch bei Workflow Management Systemen (WfMS) vorhanden, um konkrete Prozesse zu erzeugen, die dann vom WfMS gesteuert werden. Man unterscheidet zwischen Buildtime und Runtime [Ho95]. Während der Buildtime wird die Prozessdefinition mit einem Modellierungswerkzeug erstellt. Erst nach der Instanziierung durch das WfMS befindet man sich in der Runtime.

Bei EPKs wird zwischen Buildtime- und Runtime-Modellen unterschieden [ST05]. Buildtime-Modelle können im Gegensatz zu Runtime-Modellen noch nicht instanziiert werden. Sie enthalten Buildtime-Operatoren (OR_B und XOR_B), die sich von der Logik analog zu den üblichen Runtime-OR bzw. -XOR verhalten, nur mit dem Unterschied, dass sie während der Buildtime für das Schema und nicht während der Runtime, also während des Ablaufs der Prozessinstanz, entschieden werden. Erst nach Auflösung dieser Operatoren sind die Modelle Runtime-Modelle geworden, die dann instanziiert werden können.

Eine syntaktische Besonderheit besteht bei der Verwendung der Buildtime-Operatoren im Gegensatz zu den Runtime-Operatoren. In EPKs müssen unmittelbar vor den Runtime OR- und XOR-Operatoren Funktionen vorhanden sein, die die Entscheidungsfindung modellieren. Es dürfen keine Ereignisse unmittelbar vor diesen Konnektoren stehen [KNS92]. Diese Regel gilt nicht für Buildtime-Operatoren, da die dazugehörigen Entscheidungen außerhalb des Modells während der Buildtime getroffen werden. Sie können jedoch auch in die Runtime verlagert werden, indem der Buildtime-Operator durch einen Runtime-Operator mit der gleichen Logik ersetzt und eventuell eine nötige Entscheidungsfunktion unmittelbar davor hinzugefügt wird. Es gelten folgende Auflösungsregeln für die Buildtime-Operatoren nach [Sch98, ST05].

OR_B hat drei Möglichkeiten der Auflösung:

1. Auswahl der gewünschten Pfade zur Buildtime und Ersetzung des OR_B Operators durch ein Runtime AND Konnektor

2. Verlagerung in der OR-Entscheidung in die Runtime durch Hinzufügen einer Entscheidungsfunktion und eines Runtime OR-Konnektors

3. Verlagerung der Entscheidung in die Runtime durch Hinzufügen einer Entscheidungsfunktion und eines Runtime XOR-Konnektors

XOR_B hat zwei Möglichkeiten der Auflösung:

1. Auswahl eines gewünschten Pfades zur Buildtime

2. Verlagerung der XOR-Entscheidung in die Runtime durch Hinzufügen einer Entscheidungsfunktion und eines Runtime XOR-Konnektors

Die Buildtime-Operatoren werden in [Sch98, ST05] so motiviert, dass diese für Referenzmodelle (Buildtime-Modelle) verwendet werden, die für das jeweilige Unternehmen aufgelöst werden müssen. Je nachdem welche Prozessketten für das Unternehmen relevant sind, werden diese dann aus dem Buildtime-Modell ausgewählt und die nichtrelevanten ausgeschlossen. Die dort getroffenen Entscheidungen gelten für alle daraus erzeugten Prozessinstanzen. Die Buildtime-Operatoren werden damit sehr statisch verwendet im Gegensatz zu den neu eingeführten Operatoren in Kapitel 3.1.

Einen ähnlichen Ansatz zur Referenzmodellierung verfolgen die konfigurierbaren EPKs (C-EPCs) [RA05]. Konfigurierbare OR- und XOR-Konnektoren (OR^C und XOR^C) können dort mit den gleichen obigen Regeln aufgelöst werden [RA05]. Des Weiteren wurden dort konfigurierbare Funktionen eingeführt, die mit einer entsprechenden Buildtime-Entscheidung in der Prozesskette verbleibt, ausgelassen oder als optional ausführbare Funktion in die Prozesskette eingefügt wird. Zusätzlich können bei den konfigurierbaren EPKs noch Abhängigkeiten in Form von logischen Formeln an konfigurierbare Konnektoren und Funktionen annotiert werden, so dass beispielsweise bei Anwendung einer der obigen Auflösungsregeln zur Buildtime, eine konfigurierbare Funktion zwangsläufig eingefügt werden muss.

3 Operatoren für Prozessanpassungen zur Instanziierungszeit

In Abbildung 1 wurde ein Fallbeispiel als EPK vorgestellt, wie ein Teil eines Geschäftsprozesses in einer Autowerkstatt aussieht. Diese Prozessdefinitionen sind aber recht starr, so dass der Kunde vor Beginn des Geschäftsprozesses nicht noch individuelle Wünsche festlegen kann. Außerdem ist in der EPK von Abbildung 1a) eine Entscheidungsfunktion „Auftrag prüfen“ enthalten, die eigentlich nicht notwendig ist, wenn man die übliche Vorgehensweise im Unternehmen betrachtet, bei der dem ausführenden Mitarbeiter normalerweise vorher (ggf. per WfMS) mitgeteilt wird, was an dem konkreten Auto zu machen ist. Er muss damit nicht noch eine Entscheidung nach Eintreffen und Präparieren des Autos treffen (s. Entscheidungsfunktion „Auftrag prüfen“ in Abbildung 1a)), was gemacht werden muss. Vorschläge für eine realitätsnähere Modellierung mit Entscheidungsoperatoren, die vorher aufgelöst werden können, werden in diesem Abschnitt vorgestellt.

Die im Folgenden eingeführten Operatoren werden zur Instanziierungszeit aufgelöst. Diese Zeit kann die Verhandlungen zwischen Kunden und Unternehmen widerspiegeln, in der die Modalitäten für den Auftrag geklärt werden. Wir befinden uns hier noch nicht in der Runtime, da zu diesem Zeitpunkt der Auftrag noch nicht erteilt wurde, die EPK für die konkrete Instanz noch angepasst werden muss und noch keine Marken aktive Prozesse in der EPK identifizieren. Beachtet man jedoch, dass das unternehmensspezifische Modell schon in der Buildtime erzeugt wurde und dieses beispielsweise von einem WfMS bereits interpretiert wird, um die Anpassungen für die konkrete Prozessinstanz vorzunehmen, befindet man sich auch nicht mehr in der Buildtime. Man könnte die Zwischenzeit in der die Instantiationtime-Operatoren aufgelöst werden als Instantiationtime bezeichnen, die zeitlich nach der Buildtime und unmittelbar vor der Runtime liegt.

Es ergeben sich damit zwei grundlegende Konfigurationsphasen für EPKs:

1. In der Buildtime-Phase werden aus unternehmensübergreifenden Referenz-modellen durch Auflösung der Buildtime-Operatoren unternehmensspezifische Modelle erzeugt, in dem die Buildtime-Operatoren entsprechend der Regeln (s. Kapitel 2) aufgelöst werden.

2. In der Instantiationtime-Phase wird unmittelbar vor Ausführung der Modellinstanz das zugehörige Modell angepasst, jedoch ausschließlich für die zu erzeugende Prozess-Instanz. Die Instantiationtime-Operatoren werden entsprechend der im Folgenden vorgestellten Regeln aufgelöst.

Es können in Referenzmodellen jetzt also drei Sorten von Operatoren bzw. Konnektoren existieren. Buildtime-, Instantiationtime- und Runtime-Operatoren, die entsprechend ihrer Gattung entweder in einer der zwei Konfigurationsphasen vor Ausführung oder während des Ablaufs entschieden werden. Die Konfigurationsphasen können noch weiter unterteilt werden. Bei C-EPCs wird mit partiell konfigurierbaren EPKs darauf hingewiesen, dass es mehrere Konfigurationsphasen geben kann [RA05]. Eine grundlegende Differenzierung dieser Phasen wurde mit den 2 obigen vorgenommen.

3.1 Entscheidungsoperatoren für Instanziierungszeit

Die hier eingeführten Operatoren sind OR- und XOR-Operatoren, die zur Instanziierungszeit aufgelöst werden. Die Auflösungssemantik entspricht bis auf eine kleine Erweiterung (s.u.) der gleichen wie der beiden in Abschnitt 2 vorgestellten Buildtime-Operatoren. Sie brauchen des Weiteren ebenfalls keine Entscheidungsfunktionen unmittelbar vor den Operatoren, da die dazugehörigen Entscheidungen vorher in der Instanziierungszeit getroffen werden. Jedoch ist die Anwendung der Operatoren durch Auflösung in der Instanziierungszeit, welches für jede Prozessinstanz neu geschieht, abweichend von denen der Buildtime-Operatoren.

In Abbildung 2a) ist der Instantiationtime-OR-Operator (OR_I) abgebildet. Die Notation ist an die des Buildtime-Operators angelehnt. Die Unterscheidung lässt sich hier durch ein kleines I für Instantiationtime erkennen. Für die Auflösung des OR_I-Operators erscheint jedoch für das hier verwendete Fallbeispiel nur die erste der drei OR_B Auflösungsregeln aus Abschnitt 2 sinnvoll, da der Kunde dann festlegen soll, welche Leistungen er verbindlich haben möchte. Die Entscheidung dafür soll also zwingend zur Instantiationtime getroffen werden und darf in diesem Falle nicht in die Runtime verlagert werden. Diese optionale Einschränkung stellt eine Erweiterung zu den Buildtime-Operatoren dar.

Am Beispiel von Abbildung 2b) ist zu sehen, wie eine mögliche Instanz aus dem Modell aus Abbildung 1a) aussehen könnte. Aus dem Instantiationtime-OR ist in diesem Falle ein Runtime-AND geworden und der Pfeil, der zur Funktion „Auto reinigen“ führte ist gelöscht worden, so dass in diesem Fall der Kunde diese Leistung offenbar nicht erbracht haben wollte.

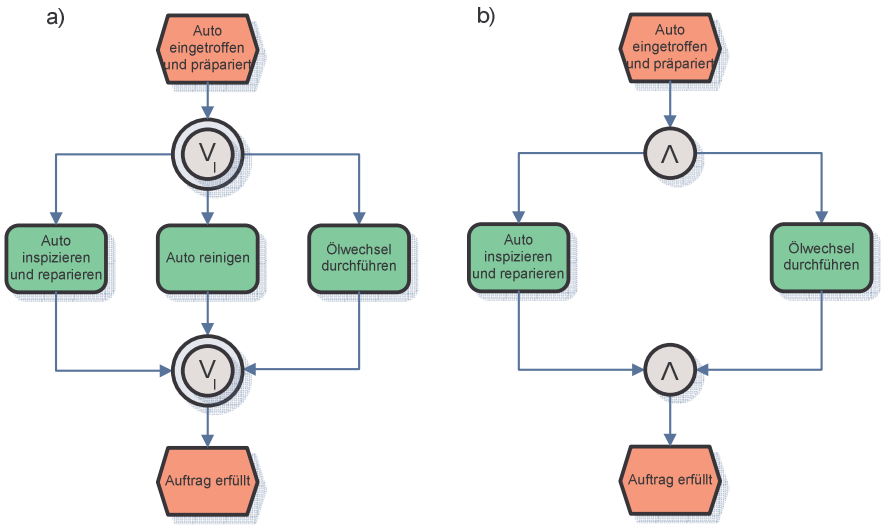

Abbildung 2: a) Instanziierungs OR-Operartor b) Modellinstanz nach Auflösung des Operators. Prozessmarken wurden weggelassen

Auch der XOR-Operator zur Instanziierungszeit hat sinnvolle Einsatzmöglichkeiten. Nehmen wir das Beispiel aus Abbildung 1b). Es könnten im Kundengespräch folgende 3 Möglichkeiten dem Kunden offeriert werden, wobei er sich zur Instanziierungszeit für eine entscheiden muss. 1.: Er wird in jedem Fall über die Schäden am Wagen benachrichtigt, 2.: Er wird ab einen bestimmten Betrag benachrichtigt, ansonsten wird automatisch repariert, 3.: Er wird nicht benachrichtigt und der Wagen wird in jedem Fall repariert.

Abbildung 3a) zeigt die entsprechende Spezifikation mit dem Instantiationtime-XOR-Operator. Bei der Instanziierung dieser EPK wird je nach Kundenwunsch ein Zweig ausgewählt. Für den Fall, dass der Kunde immer informiert werden möchte, wird bei der Instanziierung der linke Zweig ausgewählt. Die rechte Prozesskette ist für die entsprechende Prozessinstanz nicht mehr vorhanden.

Analog verhält es sich, wenn der Kunde wünscht, nicht über die Mängel informiert zu werden und damit die rechte Prozesskette wählt. Lediglich wenn der Kunde erst informiert werden möchte, wenn die Reparatur einen bestimmten Betrag überschreitet, ist die Entscheidung in die Runtime verlagert und es muss dann eine zusätzlichen Entscheidungsfunktion nach XOR_B-Auflösungsregel 2 (s. Abschnitt 2) hinzugefügt werden. Mit der Funktion „Prüfen, ob Betrag überschritten“ wird dann entschieden, ob der Kunde informiert werden muss oder nicht. Dieser Fall ist in Abbildung 3b) zu sehen.

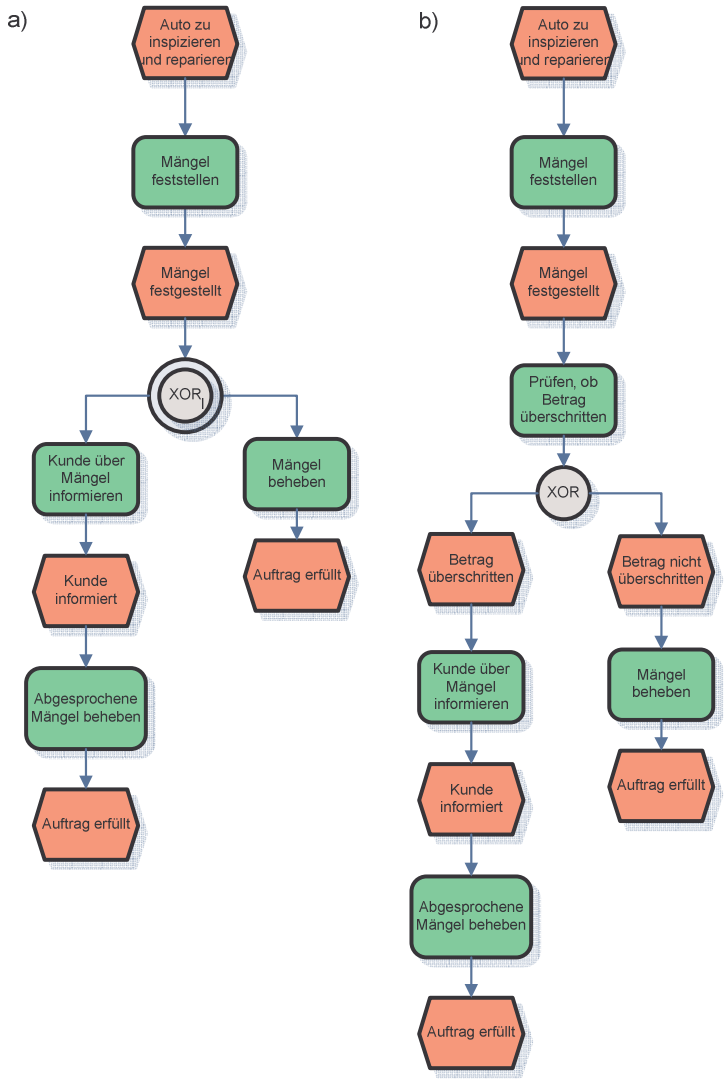

Abbildung 3: a) Instantiationtime-XOR-Operator in der Prozessdefinition. b) EPK nach Auflösung des Operators. Entscheidung ist hier in die Runtime verlegt worden.

Es erscheinen zusätzlich noch Erweiterungen für die Instantiationtime-Operatoren sinnvoll, wenn man bedenkt, dass die Modelle beispielsweise durch ein WfMS interpretiert und zur Erstellung der Prozessinstanz ausgewertet werden. Zunächst müssen die Instantiationtime-Operatoren in ein EPK-Austauschformat z.B. der EPML [MN05] eingeführt werden, damit sie von einem WfMS erkannt und interpretiert werden können. Entsprechendes wurde für die oben erwähnten C-EPCs schon gemacht [MRR05].

Bei der Modellierung in der Buildtime können Fragen mit den neuen Operatoren und Antwortalternativen für die ausgehenden Kanten verbunden werden, die im Instanziierungsprozess vom WfMS abgefragt werden. Im Beispiel von Abbildung 2a) kann z.B. die Frage: „Welche Leistungen wünscht der Kunde?“ mit dem Instantiationtime-OR verbunden werden, die vom WfMS zur Instanziierungszeit angefragt wird und die dazugehörigen Antwortalternativen mit den darauf folgenden Ereignissen oder Funktionen vorgeschlagen werden. Der Nutzer wählt die entsprechend gewünschten Leistungen aus und das WfMS löst damit den Operator auf.

Die Verwendung der neuen Instantiationtime-Operatoren erscheinen aber nicht nur für die Anpassung der EPK an die Anforderungen des Kunden sinnvoll, so wie es im bisherigen Beispiel verdeutlich wurde. Z.B. entscheidet sich auch bereits bei der Instanziierung des Prozesses, um welche Automarke bzw. –typ es sich bei dem mit dem Prozess verknüpften Auto handelt. Somit kann die EPK auch darauf bereits während der Instanziierungszeit konfiguriert werden. Eine vorzeitige bessere Ressourcenplanung wird dadurch möglich.

3.2 Operator zum Hinzufügen zusätzlicher Funktionen

Bisher wurden Operatoren besprochen, die jeweils Prozessteile zur Instanziierungszeit ausschließen können. Es sind jedoch Situationen denkbar, in denen vom Kunden speziell gewünschte Dienstleistungen noch nicht im Unternehmensangebot und entsprechend noch nicht in der EPK vorgesehen sind, diese aber trotzdem erbracht werden sollen. Um hier reagieren zu können, wird in diesem Abschnitt ein neuer Operator eingeführt, der neu benötigte Funktionen für den jeweilig zu instanziierenden Prozess geregelt hinzufügen kann. Mit Verwendung des Operators werden die Stellen in der EPK gekennzeichnet, in denen neue Prozessteile in der EPK hinzugefügt werden dürfen.

Im Folgenden werden drei Möglichkeiten vorgestellt, wie die neu hinzuzufügenden Aufgaben bzw. Funktionen in die bestehende Prozesskette eingefügt werden können. Sie können erstens unabhängig voneinander (auch parallel), zweitens abhängig voneinander in einer Sequenz und drittens alternativ zueinander angeordnet werden. Da man im voraus keine Aussage treffen kann, wie die neuen Funktionen gegenseitig und zu den bisher in der Prozessdefinition vorhandenen Funktionen in Beziehung stehen, ist es sinnvoll dem Nutzer alle 3 Möglichkeiten des Hinzufügens zur Verfügung zu stellen. Im Zweifelsfall sollte die unabhängige, parallele Modellierungsart verwendet werden. Das Modell erlaubt dann eine flexible Ausführung während der Laufzeit und damit wäre das Modellierungsprinzip „Flexibility by design“ nach [AAH08, SMR07] realisiert. Das WfMS lässt damit viele, u.U. auch nicht sinnvolle Ausführungszeitpunkte der neu hinzugefügten Aufgabe zu. Die ausführende Person hat dann die Aufgabe, den richtigen Zeitpunkt zu bestimmen.

Um das adaptive Hinzufügen der drei oben formulierten Varianten zur Instanziierungszeit zu ermöglichen, wird hier nun ein neuer ADD-Operator vorgeschlagen. Mit dem ADD-Operator werden Nutzer-gesteuert neue Funktionen und Ereignisse für die zu erzeugende Instanz der EPK eingefügt. Die Stelle des Hinzufügens der neuen Funktion wird durch die aktuelle Position des ADD-Operators in der EPK angegeben. Er ist in Abbildung 4 abgebildet und muss ebenfalls wie die Entscheidungsoperatoren aus Kapitel 3.1 zur Instanziierungszeit aufgelöst werden. Die Auflösung kann mit Hilfe eines WfMS Nutzer-gesteuert geschehen, so dass auch dieser Operator in die EPML [MN05] neu eingeführt werden muss, damit das WfMS diesen entsprechend erkennen und interpretieren kann.

Die Auflösung des ADD-Operators geschieht wie folgt. Er dient jeweils als Platzhalter für eine neu hinzuzufügende Funktion oder er kann dupliziert werden. Letztendlich müssen alle ADD-Operatoren durch konkrete Funktionen und die zugehörigen alternierenden Ereignisse ersetzt werden. Beim Duplizieren des ADD-Operators muss entschieden werden, wie der neue Operator zum vorausgehenden in Beziehung stehen soll: sequenziell, parallel/unabhängig oder alternativ. Dieses hängt davon ab, wie die neu hinzuzufügenden Aufgaben in Beziehung stehen. Hiermit wird eine Prozessstruktur während der Instanziierungszeit entstehen, die dann in eine ablauffähige EPK endet und schließlich instanziiert werden kann. Beim Erzeugen der Alternative muss dann jedoch noch eine Entscheidungsfunktion unmittelbar vor den Operator analog zur Transformation in Abbildung 3b) hinzugefügt werden. Des Weiteren muss es für den Fall, dass keine Funktionen in die bestehende Prozesskette eingefügt werden soll, möglich sein, den ADD-Operator zur Instanziierungszeit einfach zu löschen.

In Abbildung 4 ist ein Entstehungsprozess zu sehen. Ausgehend von einem ADD-Operator in a) sind bis d) drei neue Funktionen Nutzer-gesteuert hinzugefügt worden. Bei der Transformation von a) nach b) wurde der ADD-Operator unabhängig zum alten reproduziert, was mit einer vorausgehenden AND-Verknüpfung in der EPK ausgedrückt wurde. Die Beziehung der hinzuzufügenden Funktionen ist damit unabhängig und eine parallele Ausführung ist möglich. Bei der darauf folgenden Transformation nach c) hat dann gleich eine Ersetzung des linken ADD-Operators durch die Funktion „Heckklappe reparieren“ stattgefunden und zusätzlich wurde der rechte ADD-Operator in Sequenz reproduziert. In d) sind dann schließlich die übrigen ADD-Operatoren durch Funktionen ersetzt worden. Es zeigt sich dort, dass die hinzugefügten Funktionen eine Abhängigkeit besitzen, da ja zunächst die Ursache des Ölverlusts festgestellt werden muss, bevor diese behoben werden kann und daher die vorhergehend erzeugte Sequenz sinnvoll ist.

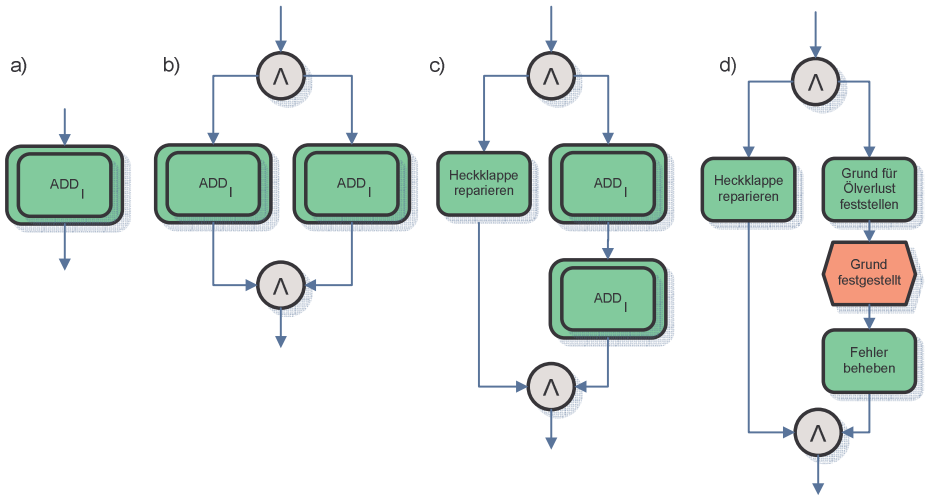

Abbildung 4: Entwicklung der Prozesskette durch den ADD-Operator zur Instanziierungszeit

3.3 Genauere Modellierung zur Instanziierungszeit

Die im Kapitel 3.2 vorgestellte Methode stellt schon viele Möglichkeiten bereit, während der Instanziierungszeit Teilprozesse individuell für die jeweilige Prozessinstanz hinzuzufügen. Es ist jedoch noch nicht möglich, die neu hinzugefügten Funktionen in die vorhandene Prozessdefinition einzuflechten.

Nehmen wir beispielsweise an, dass ein Prozess mit der in Abbildung 2a) vorgestellten EPK instanziiert werden soll und der Kunde wünscht während des Instanziierungsprozesses, insbesondere einen Ölwechsel für sein Auto zu bekommen. Zusätzlich hat dieser auch einen Ölverlust festgestellt, so dass dieser auch behoben werden soll, was dann mit dem in Kapitel 3.2 eingeführten ADD-Operator in die EPK hinzugefügt wird. Nun kann der Ölwechsel unabhängig bzw. parallel zur Reparatur der Ursache des Ölverlustes im Modell nach dem Prinzip „Flexibility by design“ eingefügt werden. Das WfMS wird in diesem Fall auch Geschäftsabläufe zulassen, die aus folgendem Grund nachteilig sind. Es kann dann nämlich der Ölwechsel vor der Reparatur der Ursache des Ölverlustes stattfinden, was zur Folge haben kann, dass das neu hinzugefügte Öl schon wieder verloren geht oder bei der Reparatur das neue Öl wieder abgelassen werden muss.

Beim obigen Geschäftsprozessbeispiel liegt es bei paralleler Modellierung der neu hinzugefügten Aufgabe in der Verantwortung des Mitarbeiters, die sinnvolle Durchführung des Prozesses zu wählen und zu erkennen, dass in dem konkreten Beispiel die Reparatur vor dem Ölwechsel zu vollziehen ist, obwohl das WfMS auch eine andere Ausführung zulässt. Eine genauere Modellierung der jeweiligen Prozessinstanz wird zwar bei korrekter Modellierung einen besseren Ablauf des Geschäftsprozesses garantieren, sie bedeutet jedoch einen bedeutend höheren Aufwand und birgt Risiken. Die editierte Workflow Definition kann Fehler enthalten, läuft dann zum ersten Mal und ist damit ungetestet, so dass im schlimmsten Fall ein Deadlock auftreten kann. Es stellt sich dadurch die Frage, inwiefern es Sinn macht, für jede Prozessinstanz den Prozess einzeln zu modellieren bzw. zu editieren.

Das genauere Modellieren des Geschäftsprozesses für die zu erzeugende Prozessinstanz kann jedenfalls durch zusätzliche Editierfunktionen der Prozessdefinition geschehen. Es können neu hinzugefügte Funktionen an andere Stellen in der Prozessdefinition verschoben werden, wo sie am sinnvollsten erscheinen. Das Verschieben muss Nutzer-gesteuert geschehen, da es in der Regel des Fachwissens bedarf, um die adaptiv neu hinzugefügten Funktionen in den bestehenden Workflow sinnvoll einzugliedern. Damit wird das Prinzip „Flexibility by change“ [AAH08, SMR07] realisiert.

Zusätzlich zum Verschieben sind noch weitere Operationen zur Veränderung bzw. Verfeinerung der Prozessdefinition denkbar. Beispielsweise können durch die Funktionsverfeinerung bei den EPKs neue Funktionen genauer spezifiziert werden. Es können Methoden aus dem Bereich des adaptiven Workflow Managements angewendet werden, in dem die Prozessdefinitionen nach [AH04] mit Hinzufügen, Ersetzen und Umstellen von Prozessen zur Laufzeit verändert werden. Weitere Änderungen sind außerdem noch bei den Change-Patterns ausgearbeitet [WR08]. Da die Mittel zur Laufzeit nach bestimmten Regeln angewendet werden dürfen, ist es natürlich insbesondere auch erlaubt, sie zum Anfang des Workflows, also zur hier untersuchten Instanziierungszeit, zu nutzen.

4 Zusammenfassung und Ausblick

Die hier vorgestellten Operatoren sollen die Prozessmodelle für die daraus zu bildenden Prozessinstanzen flexibler gestalten. Für den jeweiligen konkreten Geschäftsvorfall könnten dann z.B. Spezialwünsche des Kunden zur Instanziierungszeit in den Prozess aufgenommen oder spezielle Teile ausgeschlossen werden. Es wurden Entscheidungsoperatoren vorgestellt, die zur Instanziierungszeit für jeden neuen Geschäftsprozess entschieden werden müssen. Hierfür wurde die Idee der bereits vorhandenen Buildtime-Operatoren [Sch98] bzw. C-EPCs [RA05] verwendet. Um Prozesse des Weiteren um Funktionen zur Instanziierungszeit zu erweitern, wurde der ADD-Operator eingeführt. Die Möglichkeit des kompletten Editierens der Prozessdefinition für jede Prozessinstanz wurde anschließend in Abschnitt 3.3 diskutiert.

Für zukünftige Arbeiten ist die Umsetzung der neuen Operatoren in Verbindung mit einem Workflow Mangement System denkbar. Die EPKs müssen zunächst mit einem Modellierungstool, das die neuen Operatoren unterstützt definiert werden. Die Prozessmodelle können in einer erweiterten EPML-Codierung dem WfMS übergeben werden, das diese interpretiert. Dabei werden die Instantiationtime-Operatoren als erstes aufgelöst, indem die entsprechenden Entscheidungen bei den Instantiationtime-OR- und -XOR-Operatoren getroffen und ggf. neue Funktionen gemäß der Semantik des ADD-Operators hinzugefügt werden. Die daraus entstandenen Instanzen enthalten dann die gewünschten Funktionen, so dass die Prozessdefinition an den individuellen Anforderungen zur Instanziierungszeit angepasst wurde. Es wird an Werkzeugen zum Editieren und Anpassen von Workflowspezifikationen gearbeitet, in die die hier vorgestellten Vorschläge zu integrieren sind.

Literaturverzeichnis

[AAH08] van der Aalst, W.; Adams, M.; ter Hofstede, A.H.M.; Pesic, M.; Schonenberg, H.: Flexibility as a Service, 2008; S. 4. http://www.yawlfoundation.org/documents/BPM-08-09.pdf (gelesen: 12.10.08)

[AH04] van der Aalst, W.; van Hee, K.: Workflow Management. The MIT Press, Cambridge, 2004; S. 192 ff.

[Ho95] Hollingsworth, D.: The Workflow Reference Model, WFMC-TC-1003, 1995. http://www.wfmc.org/reference-model.html (gelesen: 12.10.08)

[KNS92] Keller, G.; Nüttgens, M.; Scheer, A.W.: Semantische Prozessmodellierung auf der Grundlage Ereignisgesteuerter Prozessketten (EPK). 1992. http://www.iwi.uni-sb.de/Download/iwihefte/heft89.pdf (gelesen: 12.10.08)

[MN05] Mendling, J.; Nüttgens, M.: EPC Markup Language (EPML) – An XML-Based Interchange Format for Event-Driven Process Chains (EPC). 2005. http://wi.wu-wien.ac.at/home/mendling/publications/TR05-EPML.pdf (gelesen: 12.10.08)

[MRR05] Mendling, J.; Recker, J.; Rosemann, M.; van der Aalst, W.: Towards the Interchange of Configurable EPCs: An XML-based Approach for Reference Model Configuration. 2005. http://wi.wu-wien.ac.at/home/mendling/publications/05-EMISA.pdf (gelesen: 12.10.08)

[NR02] Nüttgens, M.; Rump, F. J.: Syntax und Semantik Ereignisgesteuerter Prozessketten. 2002. http://ftp.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-65/06nuettgens.pdf (gelesen: 12.10.08)

[RA05] Rosemann, M.; van der Aalst, W.: A Configurable Reference Modelling Language, 2005. http://is.tm.tue.nl/staff/wvdaalst/publications/p361.pdf (gelesen: 12.10.08)

[Sch98] Schütte, R.: Grundsätze ordnungsgemäßer Referenzmodellierung: Konstruktion konfigurations- und anpassungsorientierter Modelle. Gabler, Wiesbaden, 1998, Diss. Univ. Münster; S. 246 ff.

[SMR07] Schonenberg, M.H.; Mans, R.S.; Russell, N.C.; van der Aalst, W.: Towards a Taxonomy of Process Flexibility, 2007. http://out.uclv.edu.cu/cei/Daniel%20Rojas/Towards%20a%20Taxonomy%20of%20Process%20Flexibility.pdf (gelesen: 12.10.08)

[ST05] Scheer, A.-W.; Thomas, O: Geschäftsprozessmodellierung mit der Ereignisgesteuerten Prozesskette. 2005. http://www.wiso.uni-hamburg.de/fileadmin/WISO_FS_WI/EPK-Community/Scheer_Thomas_2005_WISU_EPK.pdf (gelesen: 12.10.08)

[WR08] Weber, B.; Reichert, M.; Rinderle-Ma, S.: Change Patterns and Change Support Features – Enhancing Flexibility in Process-Aware Information Systems, 2008. http://dbis.eprints.uni-ulm.de/335/1/DKE_WRR08.pdf (gelesen: 12.10.08)