Modellierung erweiterter Beziehungen zwischen dem Start und dem Ende von Aktivitäten in Geschäftsprozessen

Szenarien, Anforderungen und Darstellungsvarianten

Thomas Bauer[1]

Abstract: Der Kontrollfluss eines Geschäftsprozesses (GP) definiert die zulässigen Ausführungsreihenfolgen von Aktivitäten. Dieser bezieht sich heutzutage jedoch ausschließlich auf gesamte Aktivitäten. Dies soll erweitert werden: Die Verwendung einzelner Start- und Endeereignisse von Aktivitäten bei der Kontrollfluss-Modellierung ermöglicht zusätzliche Ausführungsreihenfolgen. Kann beispielsweise festgelegt werden, dass das Ende der Akt. A vor dem Ende der Akt. B erfolgen muss, so erhöht dies die Flexibilität in einem Process-aware Information System (PAIS), weil die Akt. B früher gestartet werden kann, als bei einer klassischen Sequenzkante. In diesem Beitrag werden entsprechende Praxisbeispiele vorgestellt und daraus Anforderungen an die Modellierung von GP abgeleitet. Außerdem wird diskutiert, wie diese Abhängigkeiten in einem GP-Modellierungswerkzeug graphisch dargestellt werden können.

Keywords: Geschäftsprozess, Modellierung, Build-Time, Flexibilität, Kontrollfluss, Sequenz.

1 Einleitung

Ein Vorteil von PAIS [RW12] gegenüber klassischen IT-Systemen ist, dass die Einhaltung des vorgegebenen Prozesses durch das Prozess-Management-System (PMS) sichergestellt wird (Prozesssicherheit). Außerdem werden die Anwender von nicht-produktiven Aufgaben entlastet, wie dem Suchen der richtigen Programmfunktion oder der im aktuellen Geschäftsvorfall benötigten Daten. Dies erfolgt bei einem PAIS automatisch. Allerdings schränken PAIS die Flexibilität der Benutzer ein. So können sich diese durch die aktive Prozesssteuerung gegängelt fühlen. Außerdem kann der fest vorgegebene GP dazu führen, dass nicht geeignet auf Sonderfälle reagiert werden kann, obwohl hierfür betriebliche Notwendigkeit bestehen. Dadurch entstehen Nachteile für das Unternehmen. Eine Möglichkeit, um hierauf zu reagieren, sind zur Ausführungszeit (Runtime) der GP-Instanzen durchgeführte dynamische Änderungen [RW12]. Im Projekt CoPMoF (Controlable Pre-Modeled Flexibility) wird ein anderer Ansatz zur Erhöhung der Flexibilität verfolgt: Veränderungen sollen nicht dynamisch durch die Endbenutzer festgelegt werden. Stattdessen wird vorhersehbare, möglicherweise zur Runtime benötigte Flexibilität zur Buildtime vormodelliert [Ba17, Ba18]. Solche Abweichungen können geprüft und genehmigt werden, weshalb die Prozesssicherheit weiterhin gewährleistet ist. Außerdem

[1] Hochschule Neu-Ulm, Fakultät Informationsmanagement, Wileystr. 1, 89231 Neu-Ulm, thomas.bauer@hs-neu-ulm.de

doi:10.18420/inf2019_ws38

entsteht für die Endbenutzer ein deutlich geringerer Aufwand, weil alle für die Abweichung benötigten Informationen bereits zur Buildtime modelliert wurden.

In [Ba17, Ba18] wurde das Thema der vormodellierten Flexibilität bereits untersucht und viele Einzelaspekte wurden identifiziert und kurz beschrieben. Einer dieser Aspekte soll nun detailliert werden: Bisherige Prozess-Metamodelle betrachten Aktivitäten als (atomare) Einheit, zwischen denen der Kontrollfluss Reihenfolgeabhängigkeiten (z.B. eine Sequenz) definiert. Dies kann erweitert werden, indem das Start- und das Endeereignis einer Aktivität getrennt betrachtet werden. Damit können unter Anderem zusätzliche Arten von Sequenzkanten definiert werden, z.B. der Start von Akt. A muss vor dem Start der Nachfolgerakt. B erfolgen. Da hierdurch heute nicht modellierbare Ausführungsreihenfolgen entstehen, erhöht dies die Flexibilität für die Benutzer bei der GP-Ausführung (auf eine vormodellierte Art und Weise).

Dieser Ansatz zur GP-Modellierung wurde in bisherigen GP-Metamodellen und in der wissenschaftlichen Literatur kaum berücksichtigt (vgl. Abschnitt 4). Dies gilt sowohl für die semantische (d.h. fachliche) Modellierung von GP als auch für die technische Umsetzung in einem PMS (mit dem Ziel einer späteren automatischen Steuerung von Workflows). Deshalb ergibt sich folgende unbeantwortete Forschungsfrage: Welche Szenarien existieren, bei denen Abhängigkeiten mittels der Start- und Endeereignisse von Aktivitäten modelliert werden sollten, und wie kann die Modellierung solcher Abhängigkeiten in einem GP-Modellierungswerkzeug gestaltet sein?

In diesem Betrag werden die entsprechenden Anforderung dargestellt und mittels konkreter Praxisbeispiele motiviert.[2] Die Arbeit basiert auf dem Autor aufgrund seiner langjährigen Industrie- und Forschungstätigkeit im Themengebiet GP-Management bekannten Szenarien, sowie aus der Literatur bekannten bzw. offensichtlichen GP-Beispielen. Diese werden daraufhin untersucht, welche Reihenfolgeabhängigkeiten zwischen den Start- und Endeereignissen existieren. Hieraus werden abstrahierte Anforderungen abgeleitet. Aufgrund dieses Forschungsdesigns kann selbstverständlich keine Vollständigkeit der Anforderungen erreicht werden, sondern diese sind zukünftig zu erweitern. Hierfür genügt es jedoch nicht, existierende GP-Modelle nach dem Vorkommen entsprechender Modellierungskonstrukte zu durchsuchen, weil die in diesem Beitrag vorgestellten, größtenteils neuen (d.h. bisher unbekannten) Modellierungskonstrukte in existierenden GP-Modellen noch nicht verwendbar waren.

In Abschnitt 2 werden die identifizierten Anforderungen vorgestellt. Abschnitt 3 untersucht, wie entsprechende Konstrukte graphisch modelliert werden können. In Abschnitt 4 wird diskutiert, inwieweit heutige Werkzeuge, Standards und wissenschaftliche Arbeiten dieses Thema betrachten. Der Beitrag schließt mit einer Zusammenfassung und einem Ausblick.

[2] Die Erstellung einer Ausführungssemantik für solche Abhängigkeiten und ihre (prototypische) Umsetzung (von Modellierung und Ausführung) werden in diesem Beitrag noch nicht betrachtet.

2 Modellierungskonstrukte und Praxisbeispiele

Im Folgenden werden Fälle (d.h. Anforderungen) dargestellt, in denen die Reihenfolge von Aktivitäten explizit basierend auf deren Start- und Endeereignissen definiert werden sollte. Dass diese Anforderungen valide sind, wird anhand von Praxisbeispielen erläutert.

2.1 Erweiterung der Sequenz von Aktivitäten

Wie bereits erwähnt, soll das Konstrukt einer „normalen" Sequenz erweitert werden, indem alle möglichen Kombinationen von Abhängigkeiten ausgehend vom Start bzw. Ende der Vorgängeraktivität A zum Start bzw. Ende der Nachfolgeraktivität B modelliert werden können. Hierbei ergeben sich 4 mögliche Typen für eine solche Kontrollflusskante, wobei der erste Typ einer klassischen Sequenz entspricht:

1. **EndBeforeStart** Ende von Akt. A muss vor dem Start von Akt. B erfolgen
2. **EndBeforeEnd** Ende von Akt. A muss vor dem Ende von Akt. B erfolgen
3. **StartBeforeStart** Start von Akt. A muss vor dem Start von Akt. B erfolgen
4. **StartBeforeEnd** Start von Akt. A muss vor dem Ende von Akt. B erfolgen

In Abb. 1a ist dargestellt, wie solche Kontrollflusskanten modelliert werden können (alternative graphische Darstellungsmöglichkeiten werden in Abschnitt 3 diskutiert). Abb. 1b definiert jeweils die erlaubten Ausführungsreihenfolgen der Aktivitäten A und B, wobei Start(X) für den Start der Akt. X und End(X) für den Zeitpunkt von deren Beendigung steht ($X \in \{A, B\}$). Abb. 1c stellt denselben Sachverhalt graphisch dar, wobei die Zeitachse horizontal verläuft. Die Standard-Sequenzkante (1.) erlaubt am wenigsten Ausführungsmöglichkeiten (nämlich nur eine) und ermöglicht damit die geringste Flexibilität. Die 4. Variante (StartBeforeEnd) ermöglicht die meisten Ausführungsreihenfolgen (5 Stück) der Akt. A und B. Im Vergleich zu einer parallelen Ausführung (d.h. beliebige Ausführungsreihenfolgen sind erlaubt) ist lediglich die Ausführung der Akt. B vollständig vor der Akt. A ausgeschlossen (d.h. End(B) vor Start(A) ist verboten).

Eine Abhängigkeit vom Typ EndBeforeEnd (Typ 2) ist in Abb. 2 für die Akt. A und B dargestellt: die Akt. A (Fahrzeug reinigen) muss beendet werden, bevor die Akt. B (Fahrzeug zur Niederlassung des Kunden transportieren) abgeschlossen wird. Es ist nicht möglich, das Fahrzeug danach noch zu reinigen. Es ist jedoch erlaubt, dass die Akt. A und B überlappend ausgeführt werden, wenn z.B. das Fahrzeug während einer Transportpause gereinigt wird.[3]

[3] Um diesen zusätzlichen Typ 2 zu vermeiden, wäre vorstellbar, diesen Sachverhalt durch mehr Aktivitäten mit feinerer Granularität zu abzubilden (z.B. Akt. A uns B parallel, beide gefolgt von einer zusätzlichen Akt. B' „Fahrzeug übergeben"). Für eine fachliche GP-Modellierung und -Analyse ist dies akzeptabel. Bei einer GP-Steuerung durch ein PMS entsteht dadurch ein Nachteil (Zusatzaufwand für die Benutzer), weil durch die zusätzliche Akt. B' eine zusätzliche Interaktion mit der Benutzer-Arbeitsliste nötig wird.

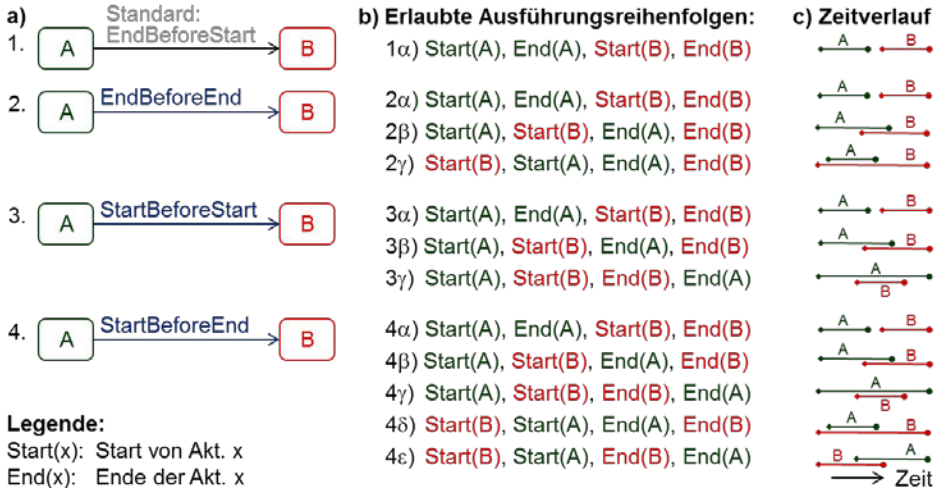

Abb. 1: Typen von Sequenzkanten zwischen den Aktivitäten A und B

Eine Beziehung vom Typ 3 (StartBeforeStart) ist für folgendes Beispiel erforderlich (vgl. Abb. 2): Die Akt. B (Fahrzeugtransport) muss beginnen, bevor die Akt. C (Kunde über anstehende Zustellung informieren) beginnt. Ansonsten, d.h. wenn die Information schon früher an den Kunden gehen würde, wäre die Gefahr einer Fehlinformation zu groß. Vor Beginn des Fahrzeugtransports ist die Wahrscheinlichkeit noch hoch, dass der Transport doch nicht stattfindet, z.B. weil der LKW nicht verfügbar oder defekt ist.

Abb. 2: Abhängigkeiten vom Typ 2 und 3 am Beispiel einer Fahrzeugauslieferung

Für eine Beziehung vom Typ 4 (StartBeforeEnd) konnten keine plausiblen Szenarien identifiziert werden. Es ist durchaus möglich, dass dieser Typ (fast) nie benötigt wird, weil er extrem viele Ausführungsreihenfolgen zulässt (vgl. Abb. 1b). Es ist aber auch möglich, dass lediglich in den betrachteten Szenarien kein entsprechendes Beispiel enthalten ist. Allerdings wird dieser Typ 4 in Kombination mit den in den nachfolgenden Abschnitten vorgestellten Funktionalitäten benötigt, z.B. zur Festlegung von Maximalzeitabständen zwischen dem Start von Akt. A und dem Ende von Akt. B (vgl. Abschnitt 2.4). Deshalb wird im Folgenden auch der Typ 4 berücksichtigt.

Prinzipiell ist es möglich, dass zwischen zwei Akt. A und B mehrere dieser Reihenfolgebeziehungen modelliert werden, die alle erfüllt sein müssen. Die erlaubten Ausführungsreihenfolgen ergeben sich als die Schnittmenge der bei den einzelnen Kantentypen erlaubten Ausführungsreihenfolgen. Allerdings macht es keinen Sinn, hierbei die die Typen 1 oder 4 zu verwenden, weil bei Typ 1 ohnehin nur eine einzige Ausführungsreihenfolge erlaubt ist (1α in Abb. 1b), und Typ 4 alle Ausführungsreihenfolgen der

anderen Typen einschließt. Somit ist ausschließlich die Kombination der Typen 2 und 3 sinnvoll. Dadurch ergeben sich als erlaubte Ausführungsreihenfolgen $2\alpha=3\alpha$ und $2\beta=3\beta$ (vgl. Abb. 1). Auch eine ODER-Verknüpfung macht nur für die Typen 2 und 3 Sinn. Die erlaubten Ausführungsreihenfolgen ergeben sich dann als die Vereinigungsmenge 2α ($=3\alpha$), 2β ($=3\beta$), 2γ und 3γ. Diese zwei Reihenfolgebeziehungen können jeweils als 2 Kanten oder als eine Kante mit 2 Typen modelliert werden, wobei auch noch die Art der Verknüpfung (AND/OR) definiert werden muss. Was hierbei besser geeignet ist, hängt von der gewählten Art der Darstellung von Sequenz-Kanten ab (vgl. Abschnitt 3).

Zwischen zwei Aktivitäten kann sogar eine zweite Sequenzkante modelliert werden, die in die umgekehrte Richtung führt. So sollte, als Erweiterung des in Abb. 2 dargestellten Beispiels, die Akt. C (Kunde informieren) abgeschlossen sein, bevor Akt. B (Fahrzeugtransport) abgeschlossen wird. Das erfordert eine zusätzliche Kante von Akt. C zu Akt. B vom Typ 2 (EndBeforeEnd). Durch beide Kanten gemeinsam wird also definiert, dass Akt. B vor Akt. C beginnen und nach C enden muss (vgl. 3γ in Abb. 1).

2.2 Optionale Sequenzkanten

Als Erweiterung der Typen 1 bis 4 sollen zudem Sequenzkanten modellierbar sein, die eine optionale Ordnung festlegen (Kanten zu Akt. C und D in Abb. 3a). Eine solche Reihenfolge soll eingehalten werden, muss aber nicht. Nach Beendigung von Akt. A wird die dann idealerweise auszuführende Akt. B den Benutzern in ihren Arbeitslisten zur Ausführung angeboten. Aber auch die optionalen Nachfolger C und D sind dort sichtbar. Allerdings sind sie entsprechend gekennzeichnet, so dass die Benutzer erkennen können, dass ihre Bearbeitung eigentlich noch nicht vorgesehen ist, sondern dass sie nur in Ausnahmefällen bereits jetzt gestartet werden sollten (vgl. Abb. 3b). Ein Benutzer kann sich also bewusst dafür entscheiden, eine dieser Aktivitäten zuerst zu bearbeiten.

Folgendes Klinikbeispiel erläutert die Notwendigkeit optionaler Sequenzkanten: Nach einer bestimmten Diagnose (Akt. A in Abb. 3a) wird bei einem Patienten üblicherweise (entsprechend dem Klinikstandard) zuerst ein EKG erstellt (Akt. B), danach ein Röntgenbild (Akt. C) und am Schluss wird die Akt. D (MRT-Aufnahme) ausgeführt. Falls aber eine der Untersuchungseinrichtungen aktuell nicht verfügbar oder überlastet ist, so kann von dieser Standardreihenfolge abgewichen werden. Alle 3 Aktivitäten befinden sich in den Arbeitslisten der Benutzer, aber die Akt. C und D sind als eigentlich noch nicht auszuführen gekennzeichnet (vgl. Abb. 3b). Ist z.B. das EKG-Gerät aktuell belegt und der Patient (z.B. Fritz Müller in Abb. 3b) wird deshalb direkt zum Röntgen geschickt, so kann der dortige Mitarbeiter die Akt. C problemlos starten. Die Akt. B und D finden später statt. Bei dem in Abb. 3a dargestellten Prozessmodell können die Aktivitäten B bis D auch überlappend ausgeführt werden, was in diesem Fall kaum sinnvoll ist, weil sich der Patient nur in einem der Untersuchungseinrichtungen befinden kann. Eine solche überlappende Ausführung kann mit einem wechselseitigen Ausschluss (vgl. Abschnitt 2.3) ausgeschlossen werden.

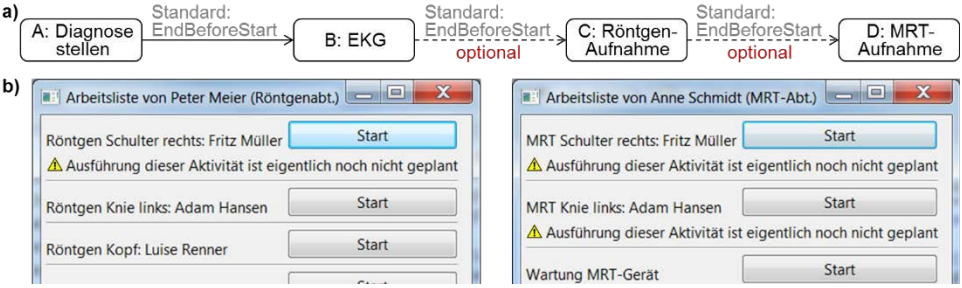

Abb. 3: a) Geschäftsprozess mit optionalen Sequenzkanten und b) Darstellung in Arbeitslisten[4]

2.3 Wechselseitiger Ausschluss (Mutual Exclusion) von Aktivitäten

Bei einem wechselseitigen Ausschluss (auch exklusiver Abschnitt genannt, engl.: Mutual Exclusion bzw. Critical Section) gilt für alle Aktivitäten des entsprechenden Bereichs (z.B. Akt. x und y), dass eine gestartete Akt. x beendet werden muss, bevor eine andere Akt. y gestartet werden kann. Es darf also zu jedem Zeitpunkt maximal eine dieser Aktivitäten ausgeführt werden. Diese Regel bezieht sich also ebenfalls auf die Start- und Ende-Ereignisse der Aktivitäten.

Alle ansonsten parallel zu bearbeitenden Aktivitäten der Mutual Exclusion werden anfänglich in den Arbeitslisten der Benutzer angeboten. Wird eine dieser Aktivitäten durch einen Benutzer zur Bearbeitung ausgewählt, so werden die anderen aus den Arbeitslisten entfernt, so dass stets nur eine Aktivität bearbeitet werden kann. Nach Abschluss dieser Aktivität erscheinen die anderen wieder in den Arbeitslisten.

Eine Mutual Exclusion lässt sich nicht vernünftig mit Sequenz-Kanten modellieren: Zwar wäre hierfür der Standard-Typ (EndBeforeStart) ausreichend (weil die Aktivitäten nacheinander bearbeitet werden). Da sich die Aktivitäten jedoch in einer Parallelität (AND-Split) befinden, sind viele unterschiedliche Ausführungsreihenfolgen möglich. Deshalb müssten alle möglichen Kombinationen von Ausführungsreihenfolgen (vgl. Abb. 4b und c) modelliert und mit XOR-Splits verknüpft werden. Dadurch wird das Prozessmodell extrem unübersichtlich. Noch problematischer ist, dass hiermit das gewünschte Verhalten nicht wirklich erreicht werden kann: Die Verzeigungsbedingungen der XOR-Splits werden ausgewertet, bevor die danach folgenden Aktivitäten in Arbeitslisten eingetragen werden. Deshalb ist es nicht möglich, dass die Auswahl eines Arbeitslisteneintrags durch einen Benutzer die nächste zu bearbeitende Aktivität festlegt. Stattdessen wird diese Entscheidung bereits beim XOR-Split (d.h. zu früh) gefällt.

[4] Für den Patienten Fritz Müller ist eigentlich Akt. B (EKG) vorgesehen. Dennoch können für ihn die Akt. C (Röntgen) und D (MRT) gestartet werden (vgl. Abb. 3b). Für Adam Hansen ist regulär die Akt. C vorgesehen, weshalb sein Röntgen-Arbeitslisteneintrag keinen Warnhinweis enthält. Auch für diesen Patienten kann die Akt. D (MRT) vorgezogen werden.

Der in Abb. 4a dargestellt Fertigungsprozess enthält eine Mutual Exclusion: Von den Akt. B, C und E darf stets nur eine einzige bearbeitet werden, d.h. die anderen werden vollständig davor oder vollständig danach ausgeführt. In diesem GP soll ein neu gegossenes Bauteil (Akt. A) zuerst gehärtet (Akt. B) und dann lackiert werden (Akt. C). Danach wird eine Rechnung hierfür erstellt (Akt. D). Parallel dazu wird das Bauteil dem Kunden vorgeführt (Akt. E). Für die Akt. B, C und E ist das Vorhandensein des Bauteils erforderlich und sie finden an unterschiedlichen Orten statt. Deshalb können sie nicht zeitlich überlappend ausgeführt werden. Dies wird mittels eines Mutual-Exclusion-Bereich modelliert. Dieser legt fest, dass keine der enthaltenen Aktivitäten gestartet werden darf, solange eine andere noch läuft. Deshalb sind nur die in Abb. 4b und c dargestellten Ausführungsreihenfolgen möglich. Da sich die Akt. D nicht im Mutual-Exclusion-Bereich befindet und lediglich nach Akt. C ausgeführt werden muss, ist im Fall α) eine mit Akt. E überlappende Ausführung erlaubt. Bei γ) wird Akt. E zwischen den Akt. B und C ausgeführt. Falls dies nicht erwünscht ist, kann diese Ausführungsreihenfolge verhindert werden, indem eine (einzige) zusammengesetzte Aktivität BC innerhalb der Mutual Exclusion modelliert wird, welche aus den Akt. B und C besteht.

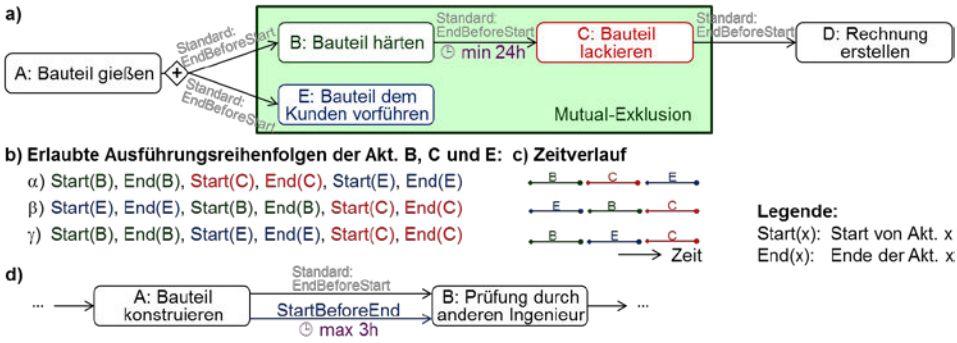

Abb. 4: Wechselseitiger Ausschluss und zeitliche Abhängigkeiten

2.4 Zeitliche Abstände zwischen Aktivitäten

Zusätzlich zur Reihenfolgebeziehung kann es erforderlich sein, dass zwischen Aktivitäten zeitliche Minimal- und Maximalabstände eingehalten werden. Bei einigen kommerziellen PMS kann bereits heute festgelegt werden, wie lange die Bearbeitung einer bestimmten (einzelnen) Aktivität maximal dauern darf. Ein Überschreiten dieser Maximalzeit führt zu einer Eskalation (z.B. Warnung an Bearbeiter, Benachrichtigung des Vorgesetzten). Dies kann verallgemeinert werden, indem Maximal- und Minimalzeiten auch zwischen unterschiedlichen Aktivitäten definiert werden können. Der Beginn und das Ende dieser Zeiträume entspricht dem Bearbeitungsbeginn (Selektion in der Arbeitsliste durch den Bearbeiter) und der Beendigung der Aktivitäten. Da dies dieselben Ereignisse sind, die auch bei den in Abschnitt 2.1 vorgestellten Kantentypen verwendet werden, können diese Zeiten als Beschriftung von solchen Sequenzkanten dargestellt

werden (vgl. Abb. 4a und d). Allerdings erfordert nicht jeder gewünschte Zeitabstand das Vorhandensein einer Reihenfolgebeziehung (Sequenz). Muss z.B. eine parallel zu Akt. X modellierte Akt. Y spätestens 3h nach dem Ende von Akt. X gestartet werden, so kann sie auch vor oder während Akt. X ausgeführt werden. Deshalb sollten Zeitabstände auch mittels „reinen Zeitkanten“ modelliert werden können, d.h. als Kanten die keine Reihenfolgebeziehung der Typen 1 bis 4 aus Abschnitt 2.1 erzwingen.

Die Relevanz von Zeitabständen soll wieder anhand von Praxisbeispielen belegt werden: In dem in Abb. 4a dargestellten Fertigungsprozess muss das in Akt. B gehärtete Bauteil mind. 24 Stunden abkühlen, bevor es lackiert (Akt. C) werden kann. Die Akt. C erscheint deswegen erst dann in den Arbeitslisten der Benutzer, wenn diese 24 Stunden verstrichen sind. Es handelt sich also um eine zeitliche Abhängigkeit zwischen dem Ende der Vorgängerakt. B und dem Start der Nachfolgerakt. C (Typ 1: EndBeforeStart).

An dieser Kante könnte (als Erweiterung von Abb. 4a) zusätzlich zu der Minimalzeit auch eine Maximalzeit modelliert werden (z.B. max. 72h). Dann ergibt sich ein Zeitintervall, in welchem die Akt. C gestartet werden muss. Dadurch wird die erforderliche Abkühlung sichergestellt, es werden aber zudem unnötige Lagerkosten für das Bauteil und inakzeptable Verzögerungen zu vermieden.

Des Weiteren können auch mehrere verschiedene Kantentypen zwischen denselben Aktivitäten modelliert werden (vgl. Abb. 4d), durch die (teilweise) Zeitrestriktionen definiert werden. So könnte in dem Fertigungsprozess aus Abb. 4a zusätzlich festgelegt werden, dass das Lackieren (Akt. B) spätestens 80 Stunden nach dem Ende der Akt. A beendet sein muss (d.h. „🕒 max. 80h“ an einer Kante vom Typ EndBeforeEnd).

Im Zusammenhang mit Zeitabständen sind auch Kanten vom Typ 4 (StartBeforeEnd) relevant. Diese legen dann evtl. gar keine Reihenfolgebeziehungen fest, wenn diese z.B. bereits restriktiver durch eine Kante vom Typ 1 (EndBeforeStart) definiert wird. Sie dienen dann lediglich zur Festlegung von Zeitabständen. So sollen bei dem in Abb. 4d dargestellten Entwicklungsprozess beide Aktivitäten innerhalb von 3 Stunden bearbeitet werden, d.h. vom Beginn der Konstruktion durch einen Ingenieur (Akt. A) bis zum Abschluss der Prüfung durch seinen Kollegen (Akt. B) dürfen maximal 3 Stunden vergehen. Dadurch werden Verzögerungen für den Gesamtprozess vermieden, aber die beiden Ingenieure können die Gesamtbearbeitungszeit flexibler nutzen (d.h. verteilen), als wenn für jede einzelne Aktivität nur 1,5 Stunden erlaubt wären.

Die zeitlichen Abstände zwischen Aktivitäten werden im Prozessmodell definiert und sind durch das PMS sicherzustellen. Allerdings kann dieses den Abschluss einer Aktivitätenbearbeitung nicht erzwingen, sondern ist auf die Mitwirkung der Benutzer angewiesen. Folgende Maßnahmen sind möglich, um die Einhaltung von Bearbeitungszeiten zu fördern bzw. zu erzwingen:

- **Mindestzeitabstand:** Ist dieser für den Start einer Nachfolgeakt. Y definiert (d.h. StartBeforeStart oder EndBeforeStart), dann wird der entsprechende Eintrag nicht in die Arbeitslisten eingestellt, bis diese Mindestzeit erreicht ist. Ein Mindestzeitab-

stand zum Ende der Akt. Y könnte erzwungen werden, indem die Beendigung der Akt. Y verhindert wird (Deaktivierung der Beendigungsfunktion). Allerdings wird es hierfür wohl kaum sinnvolle Anwendungsfälle geben.

- **Maximalabstand:** Ein solcher Zeitabstand ist sowohl für den Start als auch für die Beendigung der Nachfolgeakt. Y (dem Zielknoten der Kante) sinnvoll. Erfolgt der Start der Aktivitätenbearbeitung bzw. deren Beendigung nicht rechtzeitig, so kann eine vordefinierte Eskalation ausgelöst werden. Hierbei können dieselben Vorgehensweisen eingesetzt werden, wie heute üblicherweise zur Sicherstellung von maximalen Bearbeitungsdauern einzelner Aktivitäten (Benachrichtigung, automatische Delegation an anderen Benutzer, etc.). Um die vordefinierte Maximalzeit tatsächlich einhalten zu können, sollte die Eskalation rechtzeitig vor dem Ablaufen dieser Maximalzeit erfolgen. Auch diese „Vorwarnzeit“ sollte für die Aktivität definierbar sein.

3 Graphische Darstellung der Abhängigkeiten

Als graphische Darstellung von optionalen und zeitlichen Abhängigkeiten bietet sich eine Kantenbeschriftung an (ähnlich wie in Abb. 3 bzw. Abb. 4). Deshalb wird im Folgenden ausschließlich auf die anderen beiden Arten von Abhängigkeiten eingegangen. Hierbei werden verschiedene Darstellungsvarianten vorgestellt und ihre Vor- und Nachteile erläutert. Eine anschließende Bewertung oder gar eine empirische Untersuchung der jeweiligen Eignung, ist nicht Ziel dieses Beitrags.

Die verschiedenen Typen von Sequenzkanten (vgl. Abschnitt 2) können auf folgende Arten visualisiert werden:

1. In den Abb. 1 bis 4 wurden die Kantentypen durch ein Beschriftung unterschieden. Dies ist zwar eindeutig, aber nicht besonders „graphisch“. Hieraus resultiert der Nachteil, dass der GP-Modellierer jeden Beschriftungstext lesen und verstehen muss, d.h. dessen Bedeutung ist nicht „leicht erkennbar“.
2. Eine Alternative zu einer Beschriftung ist, für die verschiedenen Kantentypen unterschiedliche Kantenformen zu verwenden (z.B. Linie gestrichelt/doppelt, unterschiedliche Farben oder Pfeilspitzen). Dies ist sehr platzsparend, aber die Bedeutung einer Kante wird noch schwerer erkennbar.
3. Ob sich eine Kante auf das Start- oder Endeereignis einer Aktivität bezieht, kann auch dadurch symbolisiert werden, dass die Kante links oder rechts ein- bzw. ausgeht. Diese Kanten-Anknüpfungspunkte können zusätzlich durch spezielle Symbole markiert werden. In Abb. 5a wurden hierfür Kreise verwendet, die an die Start- und Ende-Events von BPMN angelehnt sind. Die Bedeutung der Kanten wird so leicht erkennbar. Allerdings ergibt sich ein etwas verwirrender Kantenverlauf, weil Kanten häufig die Umrandung von Aktivitäten schneiden.

4. In Abb. 5b und c wird der Typ der Kante durch Symbole unterschieden, die an 3. angelehnt sind. Dabei wird bei dem in Abb. 5b dargestellten Ansatz jeder Kanten-ein- und -ausgang mit einem separaten Symbol versehen, wohingegen bei Abb. 5c ein Symbol für die gesamte Kante verwendet wird. Auch bei diesem Ansatz ist der Kantentyp graphisch gut erkennbar. Da die Kanten nun wieder an beliebigen Stellen der Aktivitäten ein- und ausgehen dürfen, ergibt sich zudem ein übersichtlicher Kantenverlauf. Allerdings enthält das Diagramm zusätzliche Symbole und wird damit unübersichtlicher. Werden die Symbole für den normalen (häufigsten) Kantentyp 1 (EndBeforeStart) weggelassen, so sollte dies jedoch kaum ins Gewicht fallen.

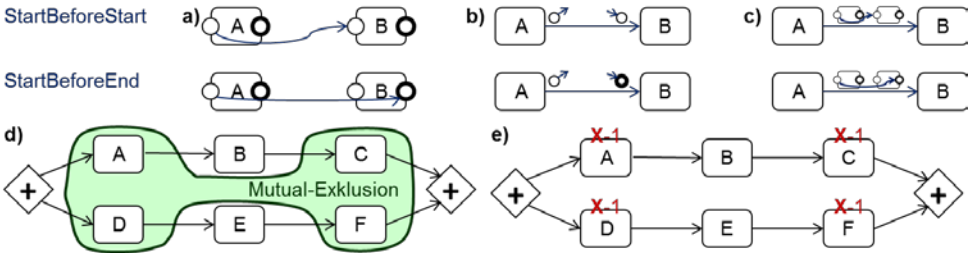

Abb. 5: Graphische Darstellung von Sequenzkanten und wechselseitigem Ausschluss

Ein wechselseitiger Ausschluss kann wie in Abb. 4a als ein farblich hinterlegter Bereich hinter den Aktivitäten symbolisiert werden. Dadurch ist er leicht erkennbar. Allerdings wird diese Darstellungsform schwer realisierbar und unübersichtlich, wenn sich nicht direkt zusammenhängende Bereiche von Aktivitäten in einem wechselseitigen Ausschluss befinden (vgl. Abb. 5d). Die Visualisierung wäre noch unübersichtlicher, wenn zwischen den beiden parallelen Pfaden noch ein dritter Pfad läge, der nicht zum wechselseitigen Ausschluss gehört.

Eine für das GP-Modellierungswerkzeug einfacher realisierbare Lösung, die zudem dieses Problem vermeidet, ist die zu einem wechselseitigen Ausschluss gehörenden Aktivitäten mit einem speziellen Symbol zu markieren. In Abb. 5e wurde hierfür ein X (wie Mutual Exclusion) gewählt. Dieses muss ggf. durch eine Nummer oder unterschiedliche Farben ergänzt werden, um verschiedene Bereiche von wechselseitigem Ausschluss zu unterschieden.

4 Stand von Forschung und Technik

Im Folgenden wird dargestellt, inwieweit die dargestellten Konzepte bereits in PMS umsetzbar sind (z.B. mittels anderer Konstrukte und Work-arounds) oder in der wissenschaftlichen Literatur betrachtet wurden.

4.1 Standards und IT-Systeme zum Geschäftsprozess-Management

Viele kommerzielle PMS basieren auf standardisierten GP-Modellierungssprachen wie BPEL [OAS07] und BPMN [OMG11]. So werden z.B. beim IBM-Produkt zur Prozesssteuerung (IBM Business Process Manager [IBM17]) BPMN-Prozesse modelliert. Bei solchen Produkten können diejenigen Konstrukte genutzt werden, die in diesen Standards enthalten sind. Die beiden Standards definieren in Bezug auf das hier betrachtete Thema weitgehend dieselben Konstrukte, die auch denen von Modellierungssprachen zur fachlichen (semantischen) GP-Modellierung (z.B. eEPKs in ARIS) ähneln.

Die erwähnten Modellierungssprachen enthalten Sequenzkanten, die jedoch ausschließlich eine rein sequentielle Bearbeitung der Aktivitäten erlauben (Typ 1). Parallelitäten ermöglichen eine zeitlich überlappende Ausführung von Aktivitäten. Hierbei ist aber jede beliebige Ausführungsreihenfolge erlaubt. Es gibt keine Konstrukte, die eine wie in Abschnitt 2.1 für die Typen 2 bis 4 vorgestellte Einschränkung ermöglichen, d.h. dass z.B. vor dem Start von Akt. B der Start von Akt. A erfolgt sein muss (StartBeforeStart). Ebenso werden keine optionalen Kontrollflusskanten angeboten.

BPEL und BPMN unterstützen das Konstrukt des wechselseitigen Ausschlusses (Mutual Exclusion, Critical Section) nicht direkt. Allerdings lässt sich diese Funktionalität z.B. in BPMN nachbilden, indem alle möglichen Ausführungsreihenfolgen modelliert werden und die XOR-Verzweigungen jeweils den Typ Deferred Choice erhalten [Ha09]. Das Konstrukt Deferred Choice wird jedoch von den meisten PMS nicht angeboten [Ha09].

In BPMN können Maximalzeiten mit einem intermediate Timer-Event [OMG11] realisiert werden. Hierzu werden zusätzliche Ausführungspfade modelliert, die bei Zeitüberschreitungen ausgeführt werden sollen. Auf das Timer-Event folgt eine speziell für diesen Zweck erstellte Eskalationsaktivität. Diese wird ausgeführt, wenn das Event eintritt (d.h. bei einer Zeitüberschreitung). Somit können maximale Zeitabstände in PMS realisiert werden, die auf BPMN basieren. Es ergeben sich hierbei jedoch recht komplexe Abläufe, mit deren Erstellung „normale GP-Modellierer“ evtl. überfordert sind.

BPMN sieht für die Visualisierung von zeitlichen Abhängigkeiten ausschließlich das Uhrensymbol für Time-Events vor. Damit sind minimale und maximale Zeitabstände zwischen Aktivitäten nicht direkt erkennbar, sondern es ergeben sich die erwähnten komplexen Prozessgraphen. Dasselbe gilt für unmittelbar auf BPMN basierende GP-Modellierungswerkzeuge. Da die anderen in Abschnitt 2 vorgestellten Abhängigkeiten in BPMN nicht vorgesehen sind, wird hierfür auch keine Visualisierung (Notation) vorgeschlagen.

4.2 Wissenschaftliche Arbeiten

Die in [RH06] vorgestellten Kontrollfluss-Pattern ermöglichen zahlreiche Ausführungsreihenfolgen von Aktivitäten. Dadurch lassen sich Abläufe so modellieren, dass die

Aktivitäten zur Runtime in einer tatsächlich gewünschten Reihenfolge ausgeführt werden. Allerdings berücksichtigen diese Pattern ausschließlich die Reihenfolge vollständig bearbeiteter Aktivitäten, z.B. Akt. A muss beendet sein bevor B gestartet werden kann. Die in Abschnitt 2.1 dargestellten Typen 2 bis 4 werden nicht berücksichtigt. Ebenso werden keine optionalen Kanten betrachtet. Diese werden in CrossFlow [Gr00, Kl00] als Optional Execution Order vorgeschlagen. Bei diesem Konstrukt sollen die Aktivitäten in der vorgegebenen Reihenfolge ausgeführt werden, dürfen aber in Ausnahmefällen auch parallel ablaufen.

Bei Constraint-basierten Ansätzen (Überblick siehe [RW12]) wird kein Kontrollflussgraph modelliert. Stattdessen werden Constraints (Regeln) festgelegt, welche die Menge der erlaubten Ausführungsreihenfolgen einschränken. Die Constraints beziehen sich, wie bei der klassischen GP-Modellierung, auf Aktivitäten als Ganzes und nicht getrennt auf deren Start- und Ende-Ereignisse. Deshalb lassen sich auch hiermit keine Sequenzen der Typen 2 bis 4 definieren. Mit optionalen Constraints [RW12] können optionale Ausführungsreihenfolgen definiert werden. Ein Constraint vom Typ respondedExistence(A, B) [RW12] ermöglicht die Festlegung eines wechselseitigen Ausschlusses für die Akt. A und B.

[He00, He01] ermöglicht die Definition beliebiger Abhängigkeiten zwischen dem Start und dem Ende von Aktivitäten (auch Typ 2 bis 4 aus Abschnitt 2.1). Bei dem Ansatz lassen sich auch wechselseitige Ausschlüsse realisieren. Allerdings ist es nicht das Ziel dieser Arbeit, Abhängigkeiten zwischen Aktivitäten derselben Prozessinstanz festzulegen, sondern zwischen Aktivitäten unterschiedlicher Prozessinstanzen und sogar unterschiedlicher Prozessvorlagen (d.h. GP-Typen). Die Modellierung solcher Abhängigkeiten ist in einem einzelnen Prozessgraph nicht möglich, weshalb sie mit hierfür zusätzlich eingeführten regulären Ausdrücken und speziellen Interaktionsgraphen definiert werden.

Auch [RH06] betrachtet den wechselseitigen Ausschluss von Aktivitäten. Allerdings werden hier nur die entsprechende Kontrollfluss-Pattern (d.h. Anforderung) Critical Section und Interleaved Routing dargestellt. Dass ein solches Modellierungskonstrukt notwendig ist, wird auch in [LK17] erläutert.

In [LWR10] werden Time-Pattern vorgestellt, welche die Festlegung von minimalen und maximalen Zeitabständen zwischen Aktivitäten erlauben. Da hierbei die Start- und Endezeitpunkte der Vor- und der Nachfolgeraktivitäten beliebig kombiniert werden können, sind alle 4 Kantentypen aus Abschnitt 2.1 abgedeckt. Allerdings werden die zeitlichen Constraints nicht im Zusammenhang mit Kontrollfluss-Kanten betrachtet.

Die graphische Darstellung eines wechselseitigen Ausschlusses (Critical Section) in [RH06] entspricht der in Abschnitt 2.3 gewählten Form. Generell betrachtet [RH06] jedoch nicht das Thema Visualisierung, sondern stellt die Funktionsweise der Kontrollfluss-Pattern dar. Constraint-basierte Ansätze verwenden keinen Prozessgraphen zur GP-Modellierung. Dementsprechend werden auch keine Vorschläge für eine geeignete Graph-Visualisierung gemacht. Die Interaktionsgraphen in [He00, He01] sind zwar graphisch, werden jedoch nicht als Modellierungstechnik für Endanwender vorge-

schlagen, sondern stellen lediglich eine formale („mathematische“) Darstellungsform für reguläre Ausdrücke dar.

5 Zusammenfassung und Ausblick

In diesem Beitrag wurden Modellierungskonstrukte eingeführt, welche die heute übliche Modellierungsmächtigkeit erweitern. Hierbei wurde anhand von konkreten Praxisbeispielen erläutert, warum diese erforderlich sind. Die durch die zusätzlichen Typen von Sequenzkanten ermöglichten Reihenfolgebeziehungen lassen sich mit klassischen Kontrollfluss-Konstrukten nicht realisieren, weil eine Modellierung als parallele Aktivitäten auch ungewünschte Ausführungsreihenfolgen ermöglichen, und normale Sequenzkanten (Typ 1) zu wenige Ausführungsreihenfolgen erlauben würde. Dadurch wäre die Flexibilität der Benutzer bei der GP-Ausführung eingeschränkt. Die Verwendung von optionalen Kontrollflusskanten und wechselseitigem Ausschluss von Aktivitäten erlaubt eine noch genauere Modellierung des gewünschten Sachverhalts. Da sich Zeitabstände, ebenso wie die neu eingeführten Typen von Sequenzkanten, auf die Start- und Endeereignisse von Aktivitäten beziehen, erlauben die Kanten zudem die Festlegung solcher zeitlicher Randbedingungen.

Dieser Beitrag stellt einen ersten Schritt in eine weitgehend neue Richtung dar. Die identifizieren Anforderungen basieren auf einer begrenzten Anzahl von GP. Um weitere Anforderungen zu identifizieren, sollten zusätzliche GP und Domänen einbezogen werden. Dabei kann evtl. auch erkannt werden, ob es relevante Praxisbeispiel für eine Sequenzkante vom Typ 4 gibt. Bzgl. der Modellierung und Visualisierung der Abhängigkeiten sollten weitere Darstellungsmöglichkeiten identifiziert werden, um z.B. besser verständliche Symbole zu entwickeln. Danach kann durch Experimente mit GP-Modellierern festgestellt werden, welche Art der Visualisierung am besten geeignet ist. Ebenso muss (nach Identifikation aller Anforderungen) eine formale Ausführungssemantik für die neuen Konstrukte entwickelt werden, damit ein PMS entsprechende GP steuern kann. Auf Basis einer (evtl. prototypischen) Implementierung können dann Fallstudien durchgeführt werden.

Als recht weitgehender Ansatz könnten Algorithmen zur Konsistenzprüfung der Prozessgraphen entwickelt werden. Dies kann nützlich sein, weil, durch die Kombination der vorgestellten Konstrukte, inkonsistente (d.h. nicht ausführbare) GP entstehen können. So führt die Kombination von 2 Sequenzkanten in dem letzten Beispiel aus Abschnitt 2.1 dazu, dass Ausführung der Akt. A die Akt. B umschließen muss (3γ in Abb. 1). Dies darf nicht mit wechselseitigem Ausschluss kombiniert werden, weil die Reihenfolge 3γ eben keine Ausführung der Aktivitäten nacheinander erlaubt. Ansonsten gibt es überhaupt keine gültige Ausführungsreihenfolge mehr. Allerdings sind solche Beispiele eher konstruiert und ein GP-Modellierer wird solche GP-Graphen in den seltensten Fällen erstellen.

Literaturverzeichnis

[Ba17] Bauer, T.: Anforderungen an vormodellierte Flexibilität für den Kontrollfluss von Geschäftsprozessen. In Proc. Informatik 2017, Workshop zum Stand, den Herausforderungen und Impulsen des Geschäftsprozessmanagements, 2017, Chemnitz; S. 799–813.

[Ba18] Bauer, T.: Vormodellierte Flexibilität für Geschäftsprozesse. In Proc. Modellierung 2018, Workshop Requirements Engineering and Business Process Management, 2018; S. 201–213.

[Gr00] Grefen, P. et al.: CrossFlow: Cross-organizational Workflow Management in Dynamic Virtual Enterprises. In Computer Systems Science & Engineering, 2000, 5; S. 277–290.

[Ha09] Havey, M.: Essential Business Process Modeling. O'Reilly, 2009.

[He00] Heinlein, C.: Workflow- und Prozeßsynchronisation mit Interaktionsausdrücken und -graphen: Konzeption und Realisierung eines Formalismus zur Spezifikation und Implementierung von Synchronisationsbedingungen. Dissertation, Universität Ulm, Fakultät für Informatik, 2000.

[He01] Heinlein, C.: Workflow and Process Synchronization with Interaction Expressions and Graphs. In Proc. 17th Int. Conf. on Data Engineering, 2001; S. 243–252.

[IBM17] IBM: Business Process Manager V8.6.0. 2017. https://www.ibm.com/support/knowledgecenter/en/SSFPJS_8.6.0, Zugriff am 25.1.2019.

[Kl00] Klingemann, J.: Controlled Flexibility in Workflow Management. In Proc. Int. Conf. on Advanced Information Systems Engineering, 2000, Stockholm; S. 126–141.

[LK17] Laue, R.; Kirchner, K.: Using Patterns for Communicating About Flexible Processes. In Proc. 18th Int. Conf. on Business Process Modeling, Development and Support, 2017, Essen; S. 12–19.

[LWR10] Lanz, A.; Weber, B.; Reichert, M.: Workflow Time Patterns for Process-Aware Information Systems. In In Proc. Enterprise, Business-Process, and Information Systems Modelling, 2010; S. 94–107.

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

[OMG11] Object Management Group: Business Process Model and Notation (BPMN) 2.0, 2011.

[RH06] Russell, N.; Hofstede, A.H.M. ter: Workflow Control-Flow Patterns. A Revised View. Universität Eindhoven, BPM Center Report 06-22, 2006.

[RW12] Reichert, M.; Weber, B.: Enabling Flexibility in Process-Aware Information Systems. Challenges, Methods, Technologies. Springer, 2012.