Vormodellierte Flexibilität für Geschäftsprozesse

Thomas Bauer [1]

Abstract: At process-aware information systems (PAIS), sometimes, a flexible deviation from the rigidly designed process becomes necessary. Otherwise the users are restricted too much. This paper presents an approach that allows to define the required flexibility only once already at build-time and realize it later in the PAIS. Compared to dynamic changes during run-time, this has the advantage that the usage of the pre-defined information reduces the effort for the end users at each deviation at run-time. In addition, applying flexibility becomes saver; e.g., since user rights can be defined. This paper presents the corresponding requirements, with a special focus on the kind of information that shall be pre-defined at build-time. Thereby, all relevant process aspects were respected and the necessity of the requirements is illustrated with examples from practice.

Zusammenfassung: Bei Process-aware Information Systems (PAIS) ist manchmal ein flexibles Abweichen vom starr modellierten Geschäftsprozess (GP) notwendig. Ansonsten werden die Benutzer zu stark eingeschränkt. Deshalb wird ein Ansatz vorgestellt, bei dem zu erwartende Flexibilität bereits zur Buildtime vormodelliert und später im PAIS umgesetzt wird. Dies hat gegenüber dynamischen Änderungen zur Runtime den Vorteil, dass, durch Nutzung der vormodellierten Informationen, bei jeder Abweichung weniger Aufwand für die Benutzer entsteht. Außerdem wird die Anwendung der Flexibilität sicherer, z.B. weil hierfür Benutzerrechte festgelegt werden können. In diesem Beitrag werden mögliche Anforderungen dargestellt, wobei insbesondere darauf eingegangen wird, welche Informationen zur Buildtime vormodelliert werden müssen. Dabei werden alle hierbei relevanten Prozessaspekte berücksichtigt und die Anforderungen werden an Praxisbeispielen erläutert.

Keywords: Geschäftsprozess, Modellierung, Buildtime, Prozessausführung, Flexibilität

CEUR-WS.org/Vol-2060/rebpm1.pdf

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 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 haben PAIS auch Nachteile: So können sich Benutzer durch die aktive Steuerung gegängelt fühlen. Außerdem kann der fest vorgegebene Prozess dazu führen, dass nicht geeignet auf Sonderfälle reagiert werden kann. Um dies zu vermeiden, muss flexibel vom modellierten GP abgewichen werden können [Re09, Sc07]. Eine spezielle Art von Flexibilität sind Abweichungen, die bereits zur Modellierungszeit (Buildtime) vormodelliert werden, damit sie zur Ausführungszeit (Runtime) angewandt werden können (Pre-Designed Flexibility [KN06], Flexibility by Design [Sc07]). In der Literatur findet sich

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

jedoch fast ausschließlich diese Kategorisierung. Eine Detaillierung der Anforderungen und von Ansätze zu deren Umsetzung waren bisher kaum Inhalt von Forschungsarbeiten.

Dieser Aspekt ist das Ziel des Projekts CoPMoF (Controllable Pre-Modelled Flexibility). Die Flexibilität in PMS soll vergrößert werden, aber Veränderungen sollen nicht willkürlich (dynamisch) durch die Benutzer festgelegt werden. Stattdessen wird vorhersehbare, möglicherweise zur Runtime benötigte Flexibilität zur Buildtime vormodelliert. Solche Abweichungen können vom GP-Modellierer detailliert spezifiziert, vom Prozessverantwortlichen genehmigt und dann implementiert (d.h. ermöglicht) werden. Dadurch ist die Prozesssicherheit weiterhin gewährleistet und Abweichungen sind nur im vorgesehenen Umfang und nur durch Benutzer mit entsprechenden Rechten möglich. Der entscheidende Vorteil ist jedoch, dass, für die Auslösung einer Abweichung ein sehr viel geringerer Aufwand entsteht, als für eine dynamische Änderung (evtl. wäre diese technisch auch gar nicht umsetzbar oder die Benutzer wären mit ihrer Definition überfordert). Angenommen, eine Überprüfung eines Entwurfes soll prinzipiell durch einen Software-Entwickler erfolgen. In schwierigen Fällen sind Entwickler hierbei jedoch überfordert. Dann soll diese Aktivität durch einen Software-Architekten desselben Projekts durchgeführt werden. Zu diesem Zweck wird bereits zur Modellierungszeit eine alternative Bearbeiterzuordnung für die Aktivität definiert und es wird festgelegt, wer diese aktivieren darf.

Dynamische Änderungen [RW12] ermöglichen z.B. das Ersetzen von Aktivitäten für eine einzelne Prozessinstanz. Für nicht vorhersehbaren Änderungen stellen sie eine unverzichtbare Funktionalität dar. Für vorhersehbare Ausnahmesituationen ist dieser Mechanismus jedoch weniger geeignet, weil bei jeder Abweichung ein Mehraufwand für die Benutzer entsteht. Im erläuterten Beispiel müsste der Benutzer eine korrekte Bearbeiterzuordnung erstellen, die (existierende) Objekte des Organisationsmodells verwenden muss. Hier ist es deutlich sinnvoller, den Aufwand ein einziges Mal bereits zur Buildtime zu betreiben, und die evtl. benötigte Bearbeiterzuordnung vorzumodellieren.

Bzgl. der Vormodellierung von Flexibilität wird in der wissenschaftlichen Literatur (wie bereits erwähnt) bisher lediglich die entsprechende Kategorie von Flexibilität definiert, diese wird jedoch nicht näher untersucht. Die einzige Ausnahme hiervon stellt der Prozessaspekt Kontrollfluss dar, für den die Vormodellierung von Flexibilität in [Ba17] beschrieben wird.[2] Damit verbleibt folgende bisher unbeantwortete Forschungsfrage: Welche Szenarien (d.h. Anforderungen) existieren für andere Prozessaspekte [Ja97], bei denen es vorteilhaft ist, zur Buildtime ein flexibles Verhalten vorzumodellieren, und welche Informationen müssen hierbei jeweils vorgegeben werden? Um die gesamte Forschungslücke zu schließen, muss ein Ansatz entwickelt werden, der all diese Anforderungen erfüllt. Im vorliegenden Beitrag wird ein Teilproblem hiervon gelöst: Es werden

[2] U.a. wurden folgende Anforderungen identifiziert. **KF-1:** Ein GP kann optionale Aktivitäten enthalten, die zwar in den Arbeitslisten erscheinen, aber vom Benutzer auch ausgelassen werden können. **KF-2:** Alternative Aktivitäten realisieren ein abweichendes Verhalten von Prozessschritten zur Behandlung von Sonderfällen. **KF-3:** Vordefinierte Vorwärts- bzw. Rückwärtssprünge ermöglichen das Auslassen bzw. Wiederholen von Prozessabschnitten, was aber vom regulären Ablauf unterscheidbar ist. **KF-4:** Bei Multi-Instanz-Parallelitäten (vgl. Akt. G und H in Abb. 1) müssen auch nach dem ∀-Split weitere Zweige hinzugefügt werden können.

unterschiedliche Geschäftsprozesse aus verschiedenen Domänen vorgestellt und daraufhin untersucht, welche Szenarien für vormodellierbare Flexibilität enthalten sind (Case Studies). Dabei wird auf diverse Einzelaspekte und -anforderungen eingegangen, um die Szenarien möglichst umfassend und verständlich darzustellen. Die Notwendigkeit der Anforderungen wird also mit Praxisbeispielen belegt. Der Kontrollfluss-Aspekt wird nicht erneut betrachtet; der Fokus liegt auf den anderen Prozessaspekten (vgl. [Ja97]). Die Entwicklung detaillierter Lösungskonzepte zur Umsetzung der Anforderungen ist kein Ziel dieses Beitrags.

Abschnitt 2 erläutert Grundlagen und die Problemstellung. In Abschnitt 3 werden Praxisbeispiele und Anforderungen beschrieben. Abschnitt 4 stellt verwandte Arbeiten vor.

2 Grundlagen und resultierende Problemstellungen

Bei der fachlichen GP-Modellierung wird, meist in einem GP-Modellierungswerkzeug, ein GP mit all seinen Anforderungen dokumentiert, so dass er später in einem IT-System als PAIS implementiert werden kann. Wenn dies mittels eines PMS erfolgt, dann wird zur Modellierungszeit (Buildtime) eine Prozessvorlage erstellt, die den später auszuführenden GP definiert. Hierzu wird u.a. ein Prozessgraph modelliert, der aus Aktivitäten besteht. Diese Prozessvorlage wird zur Ausführungszeit (Runtime) verwendet, um Prozessinstanzen zu erzeugen. Eine Prozess-Engine führt diese aus, indem für jede aktuell anstehende Aktivitäteninstanz (meist nur Aktivität genannt) entsprechende Einträge in den Arbeitslisten der potentiellen Bearbeiter erzeugt werden. Einer dieser Bearbeiter wählt einen solchen Eintrag aus, so dass er die Aktivität bearbeiten kann. Häufig besteht die Bearbeitung darin, ein Formular auszufüllen.

Der Kontrollfluss definiert die Ablaufreihenfolge mittels eines Graphen (vgl. Abb. 1). Dessen Knoten sind Aktivitäten, die von Benutzern ausgeführt werden, automatisch durchgeführte Schritte oder repräsentieren ganze Subprozesse. Der Prozessgraph enthält Gateways (Rauten), die Verzweigungen (Split) und Zusammenführung (Join) darstellen, und Schleifen. Sowohl Verzweigung als auch Schleifen stellen eine einfache Form von vormodellierter Flexibilität dar, weil sie dazu führen, dass die ausgeführten Aktivitäten und deren Ausführungsreihenfolge bei jeder Prozessinstanz unterschiedlich sein können.

Zur Realisierung des Datenflusses werden häufig Prozessvariablen verwendet. Diese sind mit den Input- und Output-Parametern der Aktivitäten verbunden, so dass ihr Inhalt beim Start der Aktivität an diese übergeben wird. Nach ihrem Ende werden die Ergebnisdaten in die Prozessvariablen (zurück-)geschrieben. Hierbei können auch komplexe Datentypen verwendet werden. So werden meist aus elementaren Datentypen zusammengesetzte Objekte und Listen (Arrays) unterstützt. Da letztere eine variable Länge haben, können sie die Basis für eine flexible Prozessausführung bilden.

Für den organisatorischen Aspekt gibt es viele Anforderungen [Ru05]. In kommerziellen PMS können üblicherweise organisatorische Objekte (z.B. Gruppe, Rolle, Abteilung) er-

zeugt und diesen Benutzer zugeordnet werden. Für jede Aktivität ist eine Bearbeiterzuordnung definiert. Dies ist eine „Formel“ (z.B. „Rolle = Software-Entwickler“), die organisatorische Objekte verwendet und mit der die Prozess-Engine die potentiellen Bearbeiter dieser Aktivität berechnet. Diese Personen erhalten dann einen entsprechenden Eintrag in ihrer Arbeitsliste oder eine eMail, so dass eine dieser Personen die Aktivität zur Bearbeitung auswählen kann. Eine gewisse Flexibilität lässt sich mittels abhängiger Bearbeiterzuordnung modellieren, weil die Aktivität dann nicht stets denselben Personen angeboten wird. So sollen die Rückfragen zu einem Dienstreiseantrag von derjenigen Person beantwortet werden, die den Antrag erstellt hat. Damit ergibt sich eine abhängige Bearbeiterzuordnung „selber Bearbeiter wie Vorgängerakt. X“.

Die Ausführung der Aktivitäten erfolgt häufig mittels Formularen. In manchen PMS sind diese ausgehend von den Input-/Output-Parametern der Aktivitäten automatisch generierbar und anpassbar. Es ist aber auch möglich, eigene Web-Formulare oder Rich-Client-Anwendungen zu realisieren, die dann über ein API an die Prozess-Engine angebunden werden. Die Ausführung automatisch ablaufender Prozessschritte realisieren viele Prozess-Engines mittels Web-Service-Aufrufen. Zur Integration von Legacy-Anwendungen werden ggf. Adapter bereitgestellt. Flexibilität ergibt sich hier lediglich aus der Vielzahl der unterstützten Applikationstypen und der Möglichkeit, bei einem Service-Aufruf einen mächtigen Enterprise-Service-Bus (ESB) zu verwenden.

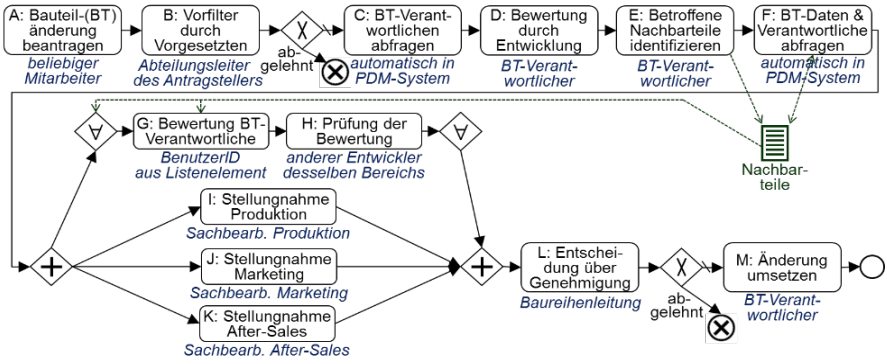

Abb. 1: Change-Management-Prozess (CMP) für Produktänderungen

Im Folgenden wird der Bedarf an Flexibilität an einem Praxisbeispiel aufgezeigt. Im Fokus liegen nicht dynamische Änderungen, mit denen auf völlig unerwartete Ereignisse reagiert wird. Stattdessen werden vorhersehbare Ausnahmefälle betrachtet. Abb. 1 zeigt einen vereinfachten Change-Management-Prozess (CMP), wie er zur Beantragung von Produktänderungen in der Automobilindustrie dient. Die Notation ist an BPMN 2.0 angelehnt. Mit der Akt. A kann ein beliebiger Mitarbeiter eine Bauteiländerung (z.B. Form der Motorhaube) beantragen. Da die Ausführung einer CMP-Instanz einen erheblichen Aufwand verursacht, kann diese in Akt. B von einer Führungskraft gestoppt werden. Akt. C ermittelt automatisch den Verantwortlichen für das zu verändernde Bauteil durch eine Abfrage an das Produktdaten-Management (PDM)-System. Dieser bewertet in Akt.

D Aufwand und Nutzen der Änderung aus Sicht des Entwicklungsbereichs. In Akt. E identifiziert er betroffene Nachbarteile (z.B. Kotflügel, Kühler), die wegen der Formänderung der Motorhaube ebenfalls angepasst werden müssen. Akt. F erfragt die zugehörigen Bauteildetails und -verantwortlichen und speichert das Ergebnis in der Liste Nachbarteile. In der Akt. G bewertet der jeweilige Verantwortliche der benachbarten Bauteile, ob die Änderung umsetzbar ist. Diese Aktivität wird mehrfach instanziiert (einmal je Nachbarteil). Dasselbe gilt für Akt. H, in der ein anderer Entwickler die Bewertung überprüft. In den Akt. I bis K erfolgt zeitgleich (parallel) eine Bewertung durch diverse Bereiche. In Akt. L wird über die Genehmigung des Änderungsantrags entschieden, und ggf. werden die Bauteile in Akt. M geändert.

Die Ausführung des CMP erfordert an einigen Stellen Flexibilität: So sind die Akt. G und H in eine Multi-Instanz-Parallelität eingebettet, d.h. die Anzahl der Zweige ergibt sich beim ∀-Split aus der Länge der Liste Nachbarteile. Diese Liste wurde von den Akt. E und F mit Input-Daten und Bearbeitern befüllt. Sie kann aber auch noch nachträglich durch eine Benutzeraktion erweitert werden. Falls ein Bauteilverantwortlicher während der Ermittlung der Nachbarteile (Akt. E) erkennt, dass er zuvor in Akt. D einen Fehler gemacht hat, so muss er diesen korrigieren können. Hierzu verändert er die Prozessvariable Bewertung, obwohl diese kein Output-Parameter von Akt. E ist. Dazu muss für diese Prozessvariable definiert sein, mit welchem Formular bzw. Tool die Änderung erfolgen kann. Für die Akt. D und E benötigt der Bauteilverantwortliche einen Stellvertreter. Da es je Bauteil nur einen Verantwortlichen gibt, kann ansonsten der gesamte Ablauf inakzeptabel verzögert werden, falls diese Person z.B. in Urlaub ist. Es genügt jedoch nicht, als Stellvertreter eine (einzige) bestimmte Person festlegen zu können. Stattdessen soll abhängig vom betroffenen Fahrzeugprojekt ein Entwickler desselben Projekts die Stellvertretung übernehmen. Bei der Erstellung der Stellungnahme der Produktion in Akt. I ist Flexibilität bzgl. der hierfür verwendeten Applikation erforderlich: Je nachdem, welcher Sachbearbeiter diese Aktivität ausführt und welche Software auf dessen Computer installiert ist, muss eine unterschiedliche Anwendung verwendet werden. So verfügen einige Benutzer über einen Viewer für CAD-Modelle, andere benutzen ein Web-Formular mit Bauteil-Bildern und manche eine „App“ auf einem Tablet, weil sie häufig unterwegs sind. Bei der Modellierung der Aktivität muss zur Buildtime also festgelegt werden, in welchen Fällen welches Applikationsprogramm aufgerufen werden soll.

3 Beispielprozesse und resultierende Anforderungen

Im Folgenden werden Anforderungen an vormodellierbare Flexibilität für unterschiedliche Prozessaspekte dargestellt. Diese betreffen sowohl die konzeptionelle Ebene (fachliche GP-Modellierung), als auch die technische Umsetzung (Ausführung) der GP.

3.1 Datenbezogener Aspekt

PMS unterstützen üblicherweise variabel lange Listen, die auch als Basis für eine Multi-Instanz-Parallelität (KF-4) dienen, um den Zweigen Anwendungsdaten und Bearbeiter zuzuordnen. Darüber hinaus ist die im Folgenden beschriebene Flexibilität erforderlich.

Prozessvariablen verändern (DF-1)

In manchen Fällen muss ein Benutzer den Inhalt einer Prozessvariablen ändern können, ohne dass hierfür eine Aktivität im Prozess vorgesehen ist. Dies ist z.B. zur Fehlerkorrektur, für Ergänzungen oder zum nachträglichen Bereitstellen von Daten nützlich.

DF-1a: Zur Buildtime muss festgelegt werden, ob eine bestimmte Prozessvariable überhaupt verändert werden darf und wer dazu berechtigt ist. Außerdem kann der Prozessbereich eingeschränkt werden, in dem eine solche Änderung möglich ist. Wurde z.B. beim CMP aus Abb. 1 in Akt. D eine fehlerhafte Bewertung eingegeben und der Prozess ist inzwischen bis zur Akt. F fortgeschritten, so wäre ein Rücksprung (vgl. KF-3) zur Akt. D und das Wiederholen der Aktivitäten ein unnötiger Aufwand. Einfacher ist eine direkte Datenkorrektur durch den Bauteilverantwortlichen. Diese ist jedoch nur erlaubt, bevor die Akt. I bis K gestartet wurden, weil diese die Bewertung als Input verwenden.

DF-1b: Falls die Bewertung aus Akt. D zusätzlich eine Skizze des Bauteils (z.B. Powerpoint-Zeichnung oder CAD-Modell) enthält und diese fehlerhaft war, so ist nicht einfach ein Text mittels eines Formulars o.ä. zu ändern. Stattdessen wird hierfür eine passende Applikation benötigt. Deshalb ist für jede (änderbare) Prozessvariable zu definieren, mit welcher Applikation eine nachträgliche Veränderung erfolgen soll.

Zuordnung von Aktivitätenparametern zu Prozessvariablen ändern (DF-2)

Auch die Abbildung von Input- und Output-Parametern der Aktivitäten auf die Prozessvariablen muss veränderbar sein. Angenommen, in dem CMP ist aktuell kein Verantwortlicher für das zu ändernde Bauteil definiert (z.B. weil dieser das Unternehmen verlassen hat). Der Bauteilverantwortliche ist aber unbedingt als Input der Akt. D und E notwendig, weil sie sonst keinem Bearbeiter zugeordnet werden können. Um dieses Problem zu lösen, soll in diesem Fall der Input-Parameter TaskBearbeiterID den Inhalt z.B. von der Prozessvariablen Bereichsleiter beziehen, anstatt von der Prozessvariablen Bauteilverantwortlicher. Hierfür ist zur Buildtime festzulegen, wer eine solche Änderung durchführen darf. Zusätzlich kann eingeschränkt werden, welche Prozessvariablen als ein bestimmter Input- bzw. Output-Parameter verwendet werden dürfen.

3.2 Organisatorischer Aspekt

Die potentiellen Bearbeiter einer Aktivität werden mittels einer Bearbeiterzuordnung berechnet. Hierfür sollen ausreichend mächtige Ausdrücke definierbar sein. Außerdem soll die Prozess-Engine diese Berechnung regelmäßig erneut ausführen (Refresh), um veränderte Zugehörigkeiten von Personen zu Rollen, Gruppen, etc. zu berücksichtigen. Ist

bei einem PMS die Funktionalität unzureichend für die benötigte Art der Bearbeiterberechnung, so ist folgender Work-around möglich: Die Bearbeiterberechnung übernimmt eine automatisch ausgeführte Vorgängeraktivität. Diese schreibt die Liste der BenutzerIDs in eine Prozessvariable, die Input-Parameter der eigentlichen Aktivität ist. Nachteilig hierbei ist, dass diese Liste nach Abschluss der Vorgängeraktivität nie mehr aktualisiert wird. Deshalb sollten folgende Anforderungen direkt vom PMS erfüllt werden.

Flexible Mechanismen zur Bearbeiterberechnung (Org-1)

Org-1a (Bearbeiter mittels Prozessdaten ermitteln): Es muss möglich sein, in einer Bearbeiterzuordnung Prozessvariablen zu verwenden. Ein Beispiel hierfür ist Akt. D in Abb. 1: Die BenutzerID des Bauteilverantwortlichen wird durch die automatisch durchgeführte Akt. C ermittelt und in einer Prozessvariablen gespeichert. Deren Inhalt wird verwendet, um Akt. D dem richtigen Bearbeiter zuzuordnen. Außer solchen BenutzerIDs können auch andere Daten für die Bearbeiterberechnung relevant sein: So kann bei der Erstellung eines Kreditantrags die betroffene Bankfiliale in eine Variable FilialID geschrieben werden. Die Aktivität „Kunde über Entscheidung informieren“ soll von einem Sachbearbeiter derselben Filiale durchgeführt werden, so dass sich als Bearbeiterzuordnung ergibt: „Rolle = Sachbearbeiter & OrgEinheit = Wert(FilialID)“.

Bei Multi-Instanz-Parallelitäten (vgl. KF-4) ist es zudem notwendig, die fortlaufende Nummer i des aktuellen Zweiges zu berücksichtigen. So soll jede Bewertung eines Bauteilverantwortlichen (Akt. G in Abb. 1) von derjenigen Person durchgeführt werden, die in der Akt. F aus dem PDM-System ermittelt und in der Liste Nachbarteile an der Indexposition i gespeichert wurde (wenn aktuell der i-te Zweig ausgeführt wird).

Die Ermittlung von BenutzerIDs durch die automatischen Akt. C bzw. F hat einen Nachteil: Ändert sich die Verantwortung für ein Bauteil nach Ausführung dieser Abfrage, so wird das nicht mehr berücksichtigt. Deshalb soll ein solcher Service-Aufruf zur Ermittlung einer BenutzerID direkt in die Bearbeiterzuordnung von Akt. D, E bzw. G integriert werden können. Diese wird in regelmäßigen Zeitabständen durch die Prozess-Engine erneut berechnet (Refresh). Das löst einen erneuten Service-Aufruf aus, so dass sich die tatsächlich aktuell gültigen potentiellen Bearbeiter ergeben.

Org-1b (Alternative Bearbeiterzuordnungen): Es kann vorhersehbar sein, dass die regulären Bearbeiterzuordnungen nicht für alle Sonderfälle geeignet sind. So kann es bei der Gruppe der regulären Bearbeiter zur Überlastungen kommen, die durch Bearbeiter aus anderen Geschäftsbereichen kompensiert werden sollen. Oder es gibt spezielle Prozessinstanzen, die von einem anderen Personenkreis bearbeitet werden sollen (z.B. weil die „normalen Bearbeiter“ hier inhaltlich überfordert wären). Hierfür sollen bereits zur Buildtime alternative Bearbeiterzuordnungen definierbar sein.

Außerdem ist bereits zur Buildtime festzulegen, wer diese Alternative aktivieren darf. Dies können reguläre Bearbeiter von Vorgängeraktivitäten, alle potentiellen Bearbeiter der betroffenen Aktivität oder aber auch Prozessverantwortliche sein. Dieses Umschalten soll auch noch möglich sein, nachdem die Aktivität in Arbeitslisten eingestellt wurde.

Dadurch wird zudem automatisch eine Neuberechnung der potentiellen Bearbeiter (Refresh) ausgelöst, so dass die alternative Bearbeiterzuordnung ab diesem Zeitpunkt auch tatsächlich wirksam wird.

Stellvertreter (Org-2)

Im CMP dürfen z.B. die Akt. D und G ausschließlich von einer einzigen Person durchgeführt werden. Ist diese nicht verfügbar, so verzögert das die gesamte Prozessausführung. Deshalb müssen für diese Aktivitäten Stellvertreter-Regelungen festgelegt werden.

Org-2a (Mittels Regeln): Zur Buildtime werden Regeln festgelegt, mit denen Stellvertreter berechnet werden. Die Regeln sollen eine ebenso mächtige Funktionalität ermöglichen, wie normale Bearbeiterzuordnungen. Insbesondere sollen auch organisatorische Objekte (z.B. Rollen, Abteilungen) verwendbar sein, so dass sich bei Änderungen im Organisationmodell automatisch aktualisierte Stellvertreter ergeben. So sollen bei Akt. D und G als Stellvertreter für den Bauteilverantwortlichen alle Personen mit „Rolle = Entwickler und derselben Projektzugehörigkeit wie der reguläre Bearbeiter“ definiert werden können. Hierbei genügt es nicht, dass eine einzelne Person als Stellvertreter festgelegt werden kann. Stattdessen werden mehrere Stellvertreter benötigt. Diese werden ggf. zu potentiellen Bearbeitern, von denen eine Person die Aktivität tatsächlich bearbeitet.

Org-2b (Abhängig vom Aktivitätentyp): Die Stellvertreter einer Person sollen abhängig von der betroffenen Aktivität festgelegt werden können. So lässt sich der Bauteilverantwortliche durch einen Kollegen desselben Projektes vertreten. Bei Projekt-unabhängigen Aufgaben (z.B. Büromaterialbestellung) vertritt ihn jedoch sein disziplinarischer Vorgesetzter. Deshalb sind Stellvertreter-Regelungen im Prozessmodell (zur Buildtime) festzulegen, d.h. für einzelne Aktivitäten(typen) oder die gesamte Prozessvorlage. Es ist nicht ausreichend, vor einer anstehenden Abwesenheit einer Person der Prozess-Engine kontextunabhängig mitzuteilen, wer dessen Stellvertreter sind.

Org-2c (Konfigurierbarkeit): Das Verhalten des Stellvertretungsmechanismus sollte konfigurierbar sein. Hierbei ist festzulegen, wann eine Stellvertretung aktiviert werden soll. Dies kann erfolgen, i) sobald ein einzelner regulärer Bearbeiter abwesend ist, ii) wenn alle ihre Stellvertretung aktiviert haben, oder iii) wenn eine bestimme Anzahl oder Quote an Abwesenden erreicht wird. Bei dieser Festlegung besteht ein Zielkonflikt zwischen der Vermeidung einer zu hohen Arbeitslast für die verbliebenen regulären Bearbeiter und dem Wunsch, dass diese die eigentlich für sie vorgesehene Aktivität auch tatsächlich bearbeiten. Außerdem ist festzulegen, ob die Aktivität mehrstufig an Stellvertreter weitergegeben werden soll. Zudem kann eingestellt werden, dass eine Aktivität, bei Rückkehr des Originalbearbeiters, den Stellvertretern wieder entzogen wird. Dies kann sich nur auf noch nicht gestartete oder auch auf bereits laufende Aktivitäten beziehen.

Die meisten dieser Anforderungen werden von vielen kommerziellen Prozess-Engines, sofern sie überhaupt Stellvertretungen anbieten, nicht erfüllt.

Benutzer-Aktionen (Org-3)

Benutzer müssen spontan Aktionen durchführen können, die den organisatorischen Aspekt betreffen. Durch diese ändert sich z.B. die Menge der (potentiellen) Bearbeiter einer Aktivität. Solche Aktionen werden auch in [Ru05] beschrieben, jedoch wird nachfolgend insbesondere betrachtet, was hierfür zur Buildtime vormodelliert werden muss.

Org-3a (Delegation): Bei einer Delegation entscheidet sich der reguläre Bearbeiter einer Aktivität, diese an andere Personen weiterzugeben, so dass sie in deren Arbeitslisten erscheint. Hierbei sollen vom PMS möglichst mächtige Mechanismen unterstützt werden. Es soll eine Delegation an mehrere Personen oder mittels einer Art Bearbeiterzuordnung möglich sein. Ein Teamleiter möchte z.B. eine Aktivität an bestimmte Teammitglieder delegieren, die im gegebenen Kontext besonders geeignet sind. Alternativ kann er die Aktivität auch an alle seine Teammitglieder delegieren. Dies erfordert besonders wenig Aufwand, wenn hierfür zur Buildtime bereits eine Regel vordefiniert wurde („Rolle = Sachbearbeiter & selbes Team wie der reguläre Bearbeiter“), die direkt zur Delegation verwendet werden kann. Zumindest muss zur Buildtime konfigurierbar sein, ob eine Delegation für eine Aktivität überhaupt erlaubt ist. Sofern dies für die Prozesssicherheit nötig ist (z.B. Einhaltung von Compliance-Regeln), soll der Personenkreis eingeschränkt werden, an den delegiert werden darf. Diese Einschränkung kann wieder durch einen organisatorischen Ausdruck, ähnlich einer Bearbeiterzuordnung, realisiert werden.

Org-3b (Potentielle Bearbeiter verändern): Die Menge der potentiellen Bearbeiter einer Aktivität wird verändert, indem Personen hinzufügt oder entfernt werden. Die anderen potentiellen Bearbeiter bleiben jedoch bestehen. Die Veränderung kann bereits erfolgen, lange bevor die Aktivität startbar ist. Hierfür ist zumindest festzulegen, wer diese Veränderung vornehmen darf. Wie bei Org-3a kann eingeschränkt werden, welche Personen überhaupt hinzugefügt werden dürfen. Auch diese Funktion ist besonders komfortabel nutzbar, wenn bereits zur Buildtime Regeln vordefiniert wurden, mittels derer Personen hinzugefügt oder entfernt werden können. Angenommen, eine Operation kann prinzipiell von jedem Arzt durchgeführt werden. Wenn der Chefarzt nun bei einem vorangehenden Patientengespräch erkennt, dass es sich um einen besonders schwierigen Fall handelt, möchte er z.B. alle Assistenzärzte als Bearbeiter ausschließen. Hierfür wurde bereits zur Buildtime die Veränderungsformel „entferne Bearbeiter mit Rolle = Assistenzarzt“ vordefiniert, die der hierfür berechtigte Chefarzt nur noch aktivieren muss.

Org-3c (Rückgabe einer Aktivität): Bearbeiter wählen Aktivitäten aus ihrer Arbeitsliste aus. Falls sie erst danach feststellen, dass sie diese Aktivität gar nicht bearbeiten möchten oder können, erfolgt eine Rückgabe. Für jede Aktivität soll konfigurierbar sein, ob eine solche Rückgabe erlaubt ist. Des Weiteren soll festgelegt werden können, bis zu welchem Zeitpunkt eine Rückgabe erfolgen kann. Für eine solche Festlegung bieten sich folgende Aktivitätenzustände an: i) Eine Rückgabe ist nur vor Beginn der Bearbeitung dieser Aktivität erlaubt. ii) Sie ist zwar nach Bearbeitungsbeginn erlaubt, aber nur solange noch kein Zwischenergebnis erzeugt wurde. Dies ermöglicht das Anschauen der Aktivitäten-Input-Daten (der Details), ohne dass Output-Daten (zwischen)gespeichert

werden. iii) Eine Rückgabe ist auch nach teilweiser Aktivitätenbearbeitung möglich, d.h. es wurden bereits Zwischenergebnisse erstellt und in Prozessvariablen der Prozess-Engine gespeichert. In diesem Fall kann zusätzlich noch festgelegt werden, ob diese verworfen werden sollen, oder ob sie für den nächsten Bearbeiter als Startzustand der Aktivität dienen, so dass dieser darauf aufbauend weiterarbeiten kann.

3.3 Ausführung von Aktivitäteninstanzen (Operationaler Aspekt)

(Kommerzielle) PMS benutzen einen Client, der Benutzern ihre Arbeitsliste dargestellt. Oft werden Formulare zur Bearbeitung der Aktivitäten verwendet. Meist werden sowohl Web-Clients als auch Rich-Clients unterstützt. Diese können über eine bereitgestellte API an die Prozess-Engine angebunden werden. Darüber hinaus bestehen die nachfolgend beschriebenen Anforderungen, die heutige PMS i.d.R. nicht vollständig erfüllen.

Unterschiedliche Applikationstypen (App-1)

Es sollen die nachfolgend dargestellten Arten von Applikationen verwendbar sein. Der Aufwand für deren Anbindung soll möglichst gering sein, d.h. sie sollten vom PMS angeboten werden und zur Buildtime nur noch geeignet konfiguriert werden müssen.

App-1a: Für die Bearbeitung von Aktivitäten darf kein Applikationstyp ausgeschlossen sein, weil die fachliche Notwendigkeit bestehen kann, hierfür eine bestimmte Anwendung zu verwenden. So kann es erforderlich sein, ein Dokument mit einem Textverarbeitungsprogramm (z.B. MS Word) oder einem CAD-Tool zu bearbeiten. Solche „Stand-alone-Applikationen“ lassen sich oft nicht in den PMS-Client integrieren. Dennoch sollten sie verwendbar sein (soweit irgendwie möglich). Hierbei ist erforderlich, dass sie automatisch gestartet und mit den richtigen Input-Daten versorgt werden. Ebenso muss ihr Ergebnis (die Output-Daten) zur PMS-Engine zurückübertragen werden.

App-1b: Es müssen auch Funktionen als Applikation verwendbar sein, die aus einem (PMS-externen) Server-System mit eigener Datenverwaltung stammen (z.B. von SAP). Dann sendet die Prozess-Engine eine Nachricht an den externen Server. Dieser stellt die Aufgabe daraufhin seinen Anwendern zur Verfügung, z.B. indem er ebenfalls Arbeitslisten anbietet. Eine Integration in die (normalen) Arbeitslisten des PMS ist oft wünschenswert. Nach Beendigung der Aktivitätenbearbeitung sendet der externe Server die Output-Daten an die Prozess-Engine zurück. Hierbei erhält die PMS-Engine evtl. nur solche Daten, die zur Ablaufsteuerung erforderlich sind (z.B. für Verzweigungsbedingungen). Die anderen verbleiben ausschließlich beim externen Server-System.

App-1c: Ein PMS soll auch mobile Clients unterstützen. Da sich entsprechende Geräte stark unterscheiden, sollen die Eigenschaften des Mobilgeräts Einfluss auf die Auswahl der potentiellen Aktivitätenbearbeiter haben (vgl. [Pr16]). So werden diese z.B. anhand des Typs des Mobilgeräts (Smartphone, Tablet, Laptop), der Bildschirmgröße, dem Aufenthaltsort oder dem aktuellem Ladezustand ausgewählt. Zur Buildtime muss dann für jede Aktivität konfigurierbar sein, welche Eigenschaften das Mobilgerät besitzen soll.

Unterschiedliche Applikationen für eine Aktivität (App-2)

App-2a: Für eine Aktivität sollen unterschiedliche Applikationen definierbar sein. Diese haben alle dieselbe Schnittstelle (d.h. Input- und Output-Daten), unterschieden sich aber in ihrem Verhalten gegenüber dem Benutzer. So kann es erforderlich sein, das Applikationsprogramm flexibel auszuwählen, z.B. abhängig von den Fähigkeiten des Benutzers (Skills), dem bevorzugten bzw. verwendeten Client-Typ (Web-/Rich-Client), der auf dem Computer des Benutzers installierten Software (z.B. MS Word, Open Office Writer) oder dem Gerätetyp (PC, Smartphone, vgl. App-1c). Die Auswahl der Applikation soll mittels Regeln (Formeln, Business Rules) erfolgen, die zur Buildtime definiert werden und Daten über den Benutzer und Prozessinstanzdaten verwenden.

App-2b: Auch noch nach dem Deployment der Prozessvorlage sollen weitere Implementierungen einer Aktivität erstellt werden können. Diese sollen natürlich auch verwendet werden, d.h. mittels Regeln nachträglich an die Prozessvorlage angebunden werden (Late Binding). Hierzu muss das Deployment der Aktivitätenimplementierung und der zugehörigen Auswahlregeln unabhängig vom eigentlichen Geschäftsprozess erfolgen. Eine entsprechende Option muss zur Buildtime auswählbar sein.

App-2c: Ein Sonderfall einer Aktivität ist eine zusammengesetzte Aktivität (Subprozess, Composed Activity). Auch für solche Aktivitäten soll es möglich sein, diese nicht fest zur Buildtime vorzugeben, sondern mittels Regeln zur Runtime auszuwählen (vgl. App-2a). Es kann wieder nötig sein, dass nach dem Deployment des (Vater-)Prozesses und dem Start der Prozessinstanz, noch weitere Subprozesse erstellt werden. Ebenso wie bei App-2b müssen die Regeln zu deren Auswahl dann nachträglich verändert werden, so dass wieder ein separates Deployment erforderlich wird.

4 Verwandte Ansätze und relevante Literatur

[KN06] stellt verschiedene Arten von Flexibilität für Geschäftsprozesse vor. Diese verfeinert [Sc07] zu „Flexibility by Design“ und „Flexibility by Underspecification“. Wie eine systematische Literaturrecherche[3] ergab, wurde bisher jedoch kaum untersucht, was zur Buildtime vormodelliert werden muss, um zur Runtime eine große Flexibilität bei geringem Aufwand für die Benutzer zu erreichen.

[RW12] schlägt Exception-Handling mit Events und Exception-Handlern vor: Für Prozessregionen wird ein (ggf. unterbrechendes) Event definiert. Falls dieses zur Runtime auftritt (Throw), wird ein Exception-Handler (Catch) ausgeführt. Dieses Vorgehen ähnelt dem in Programmiersprachen und eignet sich insb. bei technischen Fehlern wie z.B. einem Abbruch des Aktivitätenprogramms. Es kann auch zur Veränderung einer Res-

[3] Es wurde mit folgenden Suchbegriffen, jeweils in Kombination mit business process, gesucht: flexibility by design, pre-designed flexibility, flexibility buildtime, flexibility data flow, flexibility organization, flexibility activity. Außerdem wurde [RW12] als „Überblickswerk für Flexibilität in Geschäftsprozessen“ auf Hinweise auf relevante Arbeiten untersucht.

sourcen-Zuordnung (z.B. Delegation, Org-3a) verwendet werden [Ru05]. Einige Arbeiten beschäftigen sich mit Late-Binding bzw. Late-Selection (App-2). In [GBS08] wird zur Runtime mittels Regeln einer von mehreren Subprozessen ausgewählt (vgl. App-2c). Diese können unterschiedliche Realisierungen einer Aktivitätenimplementierung enthalten (App-2a). [Ad06] realisiert Aktivitäten durch sog. Worklets. Diese legen abhängig vom Kontext der Prozessinstanz fest, welche Aktivitätenimplementierung verwendet werden soll. Worklets können zur Runtime verändert werden, so dass die Applikation bzw. GUI eines Prozessschrittes anpassbar ist. Eine ähnliche Vorgehensweise verwenden kommerzielle Prozess-Engines mit Service-Orientierung. Damit bieten sie eine geeignete Basis zur Umsetzung von App-2a bis c. [WRR08] stellt sog. „Pattern for Predefined Change“ vor. Diese ermöglichen, vorzumodellieren, dass bestimmte Entscheidungen oder Festlegungen erst zur Runtime erfolgen sollen (vgl. App-2c). Case Handling [AWG05] ist ein Daten-orientierter Ansatz für wissensintensive Prozesse. Die Benutzer kennen alle Daten und können diese auch spontan ändern (vgl. DF-1). Datenänderungen erfolgen mit Formularen. Der Status einer Prozessinstanz ergibt sich aus den Inhalten der Datenobjekte. Diese definieren also, welche Aktivitäten aktuell ausführbar sind; es wird kein expliziter Kontrollfluss modelliert. Die Benutzer entscheiden selbst, welche dieser Aktivitäten ausgeführt, übersprungen oder wiederholt werden.

5 Zusammenfassung und Ausblick

In PMS muss vom starr vorgegebenen Prozess abgewichen werden können, damit diese Systeme in der Praxis benutzbar sind. Dies ist mit dynamischen Änderungen möglich, aber bei vorhersehbaren Abweichungen ein großer Aufwand und eine Fehlerquelle. Deshalb sollen vorhersehbare Sonderfälle bereits zur Buildtime vormodelliert werden. Dieser Beitrag stellt solche Szenarien und Anforderungen vor. Wünschenswert ist, dass sie zukünftig in GP-Modellierungswerkzeugen und PMS unterstützt werden, so dass entsprechende GP-Anforderungen tatsächlich dokumentiert und umgesetzt werden können.

Die Generalisierbarkeit und Relevanz der vorgestellten Szenarien sollte noch anhand von Praxisbeispielen aus anderen Domänen verifiziert werden. Außerdem sollten sie um weitere Anforderungen oder ggf. um zusätzliche Szenarien vormodellierbarer Flexibilität ergänzt werden. Dies wird dadurch erschwert, dass einige der Konstrukte bei einer klassischen Prozessmodellierung nicht vorgesehen sind (z.B. alternative Bearbeiterzuordnungen). Deshalb wurden solche Situationen in existierenden Prozessmodellen evtl. nicht erfasst, obwohl sie in der Realität eigentlich auftreten existieren. Dieses Problem sollte durch den Einsatz anderer Forschungsmethoden, wie z.B. Experteninterviews, gelöst werden. Anschließend müssen für die identifizierten Anforderungen noch detaillierte Lösungskonzepte entwickelt und ggf. prototypisch umgesetzt werden.

Literaturverzeichnis

[Ad06] Adams, M. et al.: Worklets: A Service-Oriented Implementation of Dynamic Flexibility in Workflows. In Proc. CoopIS, 2006; S. 291–308.

[AWG05] Aalst, W. van der; Weske, M.; Grünbauer, D.: Case Handling: A New Paradigm for Business Process Support. In Data & Knowledge Engineering, 2005, 53; S. 129–162.

[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.

[GBS08] Graml, T.; Bracht, R.; Spies, M.: Patterns of Business Rules to Enable Agile Business Processes. In Enterprise Information Systems, 2008, 2; S. 385–402.

[Ja97] Jablonski, S.: Architektur von Workflow-Mangement-Systemen. In Informatik Forschung und Entwicklung, Themenheft Workflow-Management, 1997, 12; S. 72–81.

[KN06] Kumar, K.; Narasipuram, M.: Defining Requirements for Business Process Flexibility. In Proc. CAiSE Workshop on Business Process Modeling, Design and Support, 2006.

[Pr16] Pryss, R. et al.: Context-Based Assignment and Execution of Human-Centric Mobile Services. In Proc. IEEE 5th Int. Conf. on Mobile Services, 2016; S. 119–126.

[Re09] Redding, G. et al.: Modelling Flexible Processes with Business Objects. In Proc. IEEE Conf. on Commerce and Enterprise Computing, 2009, Vienna; S. 41–48.

[Ru05] Russell, N. et al.: Workflow Resource Patterns: Identification, Representation and Tool Support. In Proc. CAiSE, 2005; S. 216–232.

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

[Sc07] Schonenberg, M. et al.: Towards a Taxonomy of Process Flexibility (Extended Version), Eindhoven University of Technology, 2007.

[WRR08] Weber, B.; Reichert, M.; Rinderle-Ma, S.: Change Patterns and Change Support Features. In Data and Knowledge Engineering, 2008, 66; S. 438–466.