Prozessmanagementsysteme

Nur ein wenig Flexibilität wird nicht reichen

Peter Dadam · Manfred Reichert
Stefanie Rinderle-Ma

Einführung

Die zunehmende Verschärfung des Wettbewerbs sowie neue Wettbewerbsformen erzeugen für alle Unternehmen einen starken Druck, sich immer rascher auf neue Marktsituationen einstellen zu müssen. So kann es erforderlich werden, dass sie ihre Produkte und Dienstleistungen in immer kürzerer Zeit entwickeln und herstellen, dass sie diese mit geringeren Kosten realisieren, dass sie neue Formen der Interaktion mit ihren Kunden entwickeln oder dass sie ihre Produkte in Kooperation mit anderen Partnern entwickeln oder herstellen müssen.

Diese Entwicklungen bedeuten, dass ein Unternehmen in der Lage sein muss, seine betrieblichen Abläufe, d. h. seine (Geschäfts-)Prozesse, sehr rasch an neue Gegebenheiten anzupassen, wenn es weiter im Wettbewerb bestehen will. Dies haben auch die meisten Unternehmen inzwischen erkannt. Wenn man den diversen Umfragen und Prognosen Glauben schenken darf, rangiert das Thema *Business Process Management* (BPM) [19] in der Prioritätenskala der IT-Verantwortlichen mittlerweile ganz weit oben. Die bislang in den Unternehmen im Wesentlichen immer noch überwiegend manuell und papiergebunden durchgeführten betrieblichen Abläufe (Prozesse) werden zunehmend als fehlerträchtig, kostenintensiv und schwerfällig hinsichtlich Änderungen angesehen. Wesentliche Gründe für die Einführung von *Prozessmanagementsystemen* sind unserer Erfahrung nach der Wunsch bzw. die Notwendigkeit nach einer

- „aktiven" Prozessunterstützung, indem den Bearbeitern die für sie bestimmten Aufgaben in deren Arbeitslisten gestellt werden, sobald diese Aufgaben ausführbar sind (Abb. 1),
- Automatisierung von Teilaufgaben bzw. Teilprozessen,
- Integration heterogener Anwendungssysteme,
- rascheren Anpassbarkeit der betrieblichen Abläufe an neue Gegebenheiten und
- besseren Nachvollziehbarkeit einzelner Prozessausführungen, z. B. im Zusammenhang mit der Erfüllung von Compliance-Vorgaben.

Welche funktionalen Anforderungen ein *Prozessmanagementsystem* (PMS) erfüllen muss, hängt stark davon ab, welche Art von Anwendungsfunktionen ausführbar sein sollen. Relativ geringe Anforderungen haben diesbezüglich PMS, die im Wesentlichen lediglich den Inhalt von Datenbanktabellen anzeigen und manipulieren lassen (*formularorientierte PMS*) oder die sich auf das Anzeigen und Bearbeiten von Dokumenten (*dokumentenorientierte PMS*) beschränken. Im Allgemeinen wird man jedoch von einem PMS erwarten, dass mit den Prozessschritten beliebige Anwendungsfunktionen verknüpft werden können. Dies können im Prinzip Dienste („Services") jeglicher Art sein, also z. B. Formular-

DOI 10.1007/s00287-010-0456-0
© Springer-Verlag 2010

Peter Dadam · Manfred Reichert
Universität Ulm, Institut für Datenbanken und Informationssysteme (DBIS),
James-Franck-Ring, 89069 Ulm, Deutschland
E-Mail: {peter.dadam, manfred.reichert}@uni-ulm.de

Stefanie Rinderle-Ma
Universität Wien, Fakultät für Informatik,
Workflow Systems and Technology,
Rathausstraße 19/9, 1010 Wien, Österreich
E-Mail: stefanie.rinderle-ma@univie.ac.at

Published online: 09 July 2010

Zusammenfassung

Die heute angebotenen Prozess- und Workflowmanagementsysteme basieren weitgehend auf der Annahme, dass sich die zu unterstützenden Arbeitsabläufe – inklusive Ausnahmen – komplett vormodellieren lassen. Ein Abweichen vom vorgeplanten Ablauf zur Laufzeit ist, falls überhaupt vorgesehen, nur unter sehr starken Einschränkungen möglich. Für Anwendungsumgebungen, die eine rasche und flexible Reaktion auf nicht vorhergesehene Ausnahmesituationen und Notfälle erfordern, sind Systeme dieser Art praktisch jedoch nicht einsetzbar. Der Beitrag beschreibt, welche grundlegenden technologischen Eigenschaften ein Prozessmanagementsystem erfüllen muss, um den Flexibilitätsanforderungen der realen Welt gerecht zu werden.

und Dokumentbearbeitungsfunktionen, Datenbankaufrufe, Aufrufe von externen Anwendungen, das Versenden/Empfangen von E-Mails oder der Aufruf von Webservices zur Abfrage von Wetterdaten oder Börsenkursen. Wesentlich bei solch allgemein einsetzbaren PMS ist, dass im Allgemeinen Dienste gerufen werden, denen Aufrufparameter übergeben und von denen Resultatparameter entgegengenommen werden müssen. Diese Resultatparameter können wiederum Eingabeparameter für nachfolgende Prozessschritte darstellen, können aber auch Einfluss auf den sog. Kontrollfluss des Prozesses nehmen, indem sie Werte für Verzweigungsentscheidungen oder Schleifenbedingungen liefern.

Zu Beginn einer „Elektronifizierung“ der Prozesse eines Unternehmens werden in der Regel elementare Arbeitsabläufe im Vordergrund stehen, wie z. B. Posteingangsbearbeitung oder Reisekostenabrechnung. Hat sich diese Maßnahme bewährt, wird jedoch bald der Wunsch aufkommen, auch anspruchsvollere Prozesse zu unterstützen, wie z. B. Entwicklungsprozesse für Industrieprodukte, Prozesse im Rahmen von Bauprojekten oder Behandlungsprozesse im Krankenhaus. Bei solchen, aber auch schon bei weit weniger anspruchsvollen Prozessen lassen sich zur Entwurfszeit nicht mehr alle möglichen Ausnahme- und Sonderfälle antizipieren und vormodellieren. Eine naive Anwendung heutiger PMS in diesen Bereichen könnte deshalb entweder bedeuten, dass man in Sonderfällen bzw. Ausnahmesituationen nicht mehr flexibel reagieren kann oder aber das Problem unter Umgehung des PMS lösen muss. Die erste Alternative dürfte meist nicht akzeptabel sein und die zweite wäre unter dem Aspekt der Nachvollziehbarkeit und Fehlervermeidung ebenfalls keine adäquate Lösung.

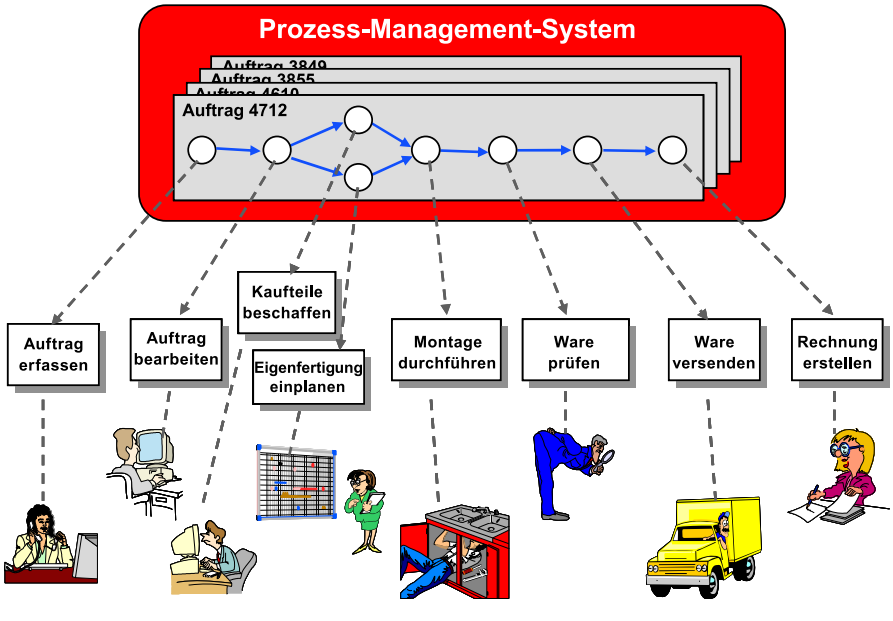


Abb. 1 Ausführen von Prozessen durch ein Prozessmanagementsystem

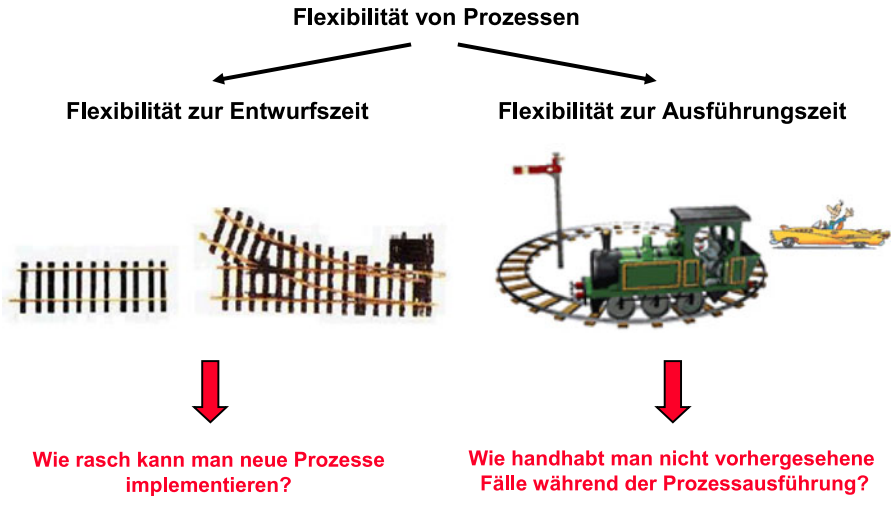


Abb. 2 Arten von „Flexibilität" von Prozessen

Möchte man eine durchgängige Prozessunterstützung realisieren, die von den elementaren Arbeitsabläufen bis hin zu anspruchsvollen Entscheidungsprozessen reicht, benötigt man PMS, die erheblich flexibler sind als die derzeit gängigen Systeme. Welche technologischen Fähigkeiten man hierzu benötigt, und wie eine solche Technologie aussehen könnte, wird in den nachfolgenden Kapiteln diskutiert und exemplarisch beschrieben.

Arten von Flexibilität

Wie eingangs erwähnt, ist eines der wesentlichen Motive für die Einführung von PMS die Erhöhung der Flexibilität des Unternehmens. Deshalb wirbt praktisch auch jeder PMS-Anbieter damit, dass dies mit seiner Prozessmanagementtechnologie erreicht werden kann. Hierbei wird allerdings unterschlagen, dass man zwei Arten von Flexibilität in Bezug auf Prozesse unterscheiden muss – *Flexibilität zur Entwurfszeit* und *Flexibilität zur Ausführungszeit* (Abb. 2) und viele Anwender haben derzeit auch noch realisiert, dass unzureichende oder gar überhaupt nicht vorhandene Flexibilität bei der Prozessausführung zu großen Problemen führen kann. Wir werden beide Arten von Flexibilität nachfolgend diskutieren. Ein weiterer Flexibilitätsaspekt ist die *Prozesschemaevolution*, auf die wir ebenfalls eingehen werden.

Flexibilität zur Entwurfszeit

Flexibilität zur Entwurfszeit bedeutet, dass man rasch neue Prozessmodelle erstellen kann. Moderne PMS bieten hierfür in der Regel grafische Modellierungswerkzeuge an, mittels derer man die Prozesse beschreiben und die zur Verfügung stehenden Anwendungsfunktionen mittels Drag and Drop oder Kontextmenü in die Prozessmodelle einfügen kann. Dieser Aspekt ist bei modernen PMS in der Regel recht gut gelöst, d. h. man kann auf diese Weise relativ rasch neue Prozesse komponieren.

Ein ganz wesentlicher Aspekt in diesem Zusammenhang ist allerdings, wie rasch man ein solches Prozessmodell anschließend „in Produktion" bringen kann. Das heißt, wie viel Aufwand ist erforderlich, um herauszufinden, ob der erstellte Prozess technisch robust und stabil läuft, d. h. keine Laufzeitfehler verursachen wird. Diesbezüglich sieht es bei heutigen PMS schon sehr viel schlechter aus. Die systemseitige Fehleranalyse des Prozessmodells ist meist recht schwach ausgeprägt, sodass die Prozessmodellierer nur durch intensives Testen bzw. Durchspielen aller möglichen Prozessausführungen herausfinden können, ob der Prozess technisch stabil und robust ausführbar sein wird. Einer der Gründe für diese oftmals schwache Unterstützung ist, dass viele Hersteller bei der Realisierung ihres PMS dachten, dass eine Prozessmodellierungssprache dann besonders gut und praxistauglich ist, wenn sie viele Modellkonstrukte anbietet und auch in Bezug auf die Strukturierung des Prozessmodells möglichst viele Freiheiten zulässt. Zeitweise gab es sogar einen regelrechten Wettlauf, wer mehr Ablaufmuster (Workflow-Patterns; siehe [20]) in seiner Modellierungssprache direkt abbilden kann. Eine Folge sehr liberaler Prozessmodelle ist jedoch, dass es oftmals systemseitig nicht möglich

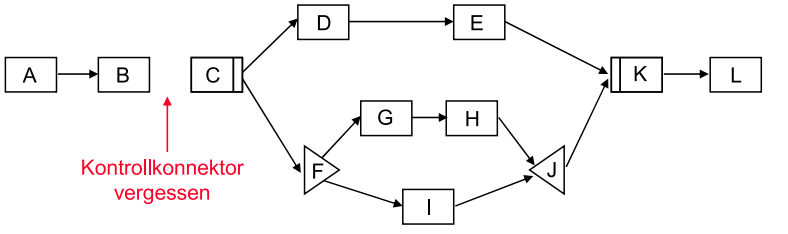


Abb. 3 Modellierungsfehler – fehlender Kontrollkonnektor

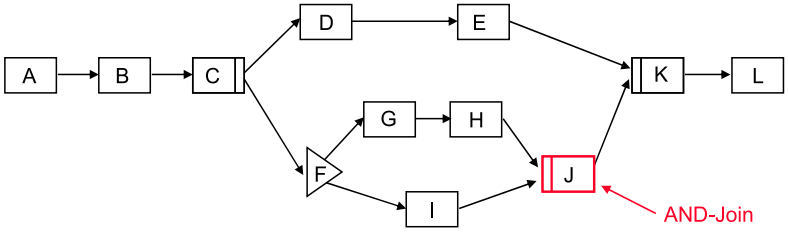


Abb. 4 Modellierungsfehler – XOR-Split mit AND-Join abgeschlossen

ist, Modellierungsfehler als solche überhaupt noch zu erkennen [7, 9].

Zur Veranschaulichung betrachten wir zwei einfache Beispiele. Wenn ein PMS erlaubt, dass ein Prozessmodell mehrere Startknoten haben kann, würde die Ausführung des in Abb. 3 dargestellten Prozesses parallel mit den Schritten A und C beginnen und die beiden abgebildeten Fragmente würden als quasi voneinander unabhängige Teilprozesse ausgeführt werden. Dies hat der Prozessmodellierer aber ggf. nicht beabsichtigt, was zu Inkonsistenzen oder Laufzeitfehlern führen kann. In diesem einfachen Beispiel sieht man einen solchen Modellierungsfehler natürlich sofort. In einem großen, komplexen Prozessmodell mit dutzenden oder gar hunderten von Prozessschritten hingegen kann die Suche nach Fehlern dieser Art aber schnell zu einem Alptraum werden.

In dem in Abb. 4 dargestellten Beispiel habe der Modellierer versehentlich einen XOR-Split (d. h. eine alternative Verzweigung bei Schritt F) mit einem AND-Join abgeschlossen. Falls ein PMS eine solche Konstellation zulässt bzw. nicht als Fehler erkennt, hängt es von der konkret implementierten Schaltsemantik im PMS ab, was passiert, wenn die Ausführung der Prozessinstanz am Knoten J ankommt. Vom Laufzeitfehler über Blockierung bis zu frühzeitiger Terminierung der Prozessinstanz (ohne Fehlermeldung) ist hier im Prinzip alles denkbar.

Weitere Probleme ergeben sich, wenn die Prozessmodellierungssprache auch Schleifen erlaubt, es hierfür aber kein spezielles Schleifenkonstrukt gibt. In diesem Fall kann man einen versehentlich modellierten Zyklus, der möglicherweise zu einer Blockierung führt, nicht von einer gewollten Schleife unterscheiden.

Leider ist in heutigen PMS die mangelnde Unterstützung zur Erkennung oder Vermeidung von Fehlern der obigen Art nicht der einzige Grund, warum nach der Modellierung meist ausführliches Debuggen und Testen der Prozesse angesagt ist. Ebenso kritisch ist, dass die meisten PMS nicht wissen, welche Datenflüsse zwischen den Anwendungsfunktionen stattfinden (oder aber dieses Wissen nicht ausnutzen). Das heißt, das PMS weiß nicht, welche Parameter beim Aufruf versorgt bzw. welche Eingabedaten vorhanden sein müssen und welche Rückgabewerte nach dem Aufruf zurückkommen (Abb. 5). Dieses Manko hat zur Folge, dass ein solches PMS zur Modellierungszeit auch nicht prüfen kann, ob die durch das Prozessmodell (d. h. den „Kontrollfluss“) festgelegten möglichen Aufrufreihenfolgen der Anwendungsfunktionen mit den Datenflüssen zwischen ihnen harmonieren. Wenn diese beiden Aspekte in Widerspruch zueinander stehen, werden möglicherweise Anwendungsfunktionen aufgerufen, deren Eingabedaten nicht zur Verfügung stehen. Dies kann zu einem Laufzeitfehler dieser Anwendungsfunktion, etwa in Form eines Programmabsturzes, führen.

Flexibilität zur Ausführungszeit

Flexibilität zur Ausführungszeit bedeutet, dass man auf Prozessinstanzebene (z. B. bei der Bearbeitung der Kreditanfrage eines bestimmten Kunden oder der Behandlung eines bestimmten Patienten) vom

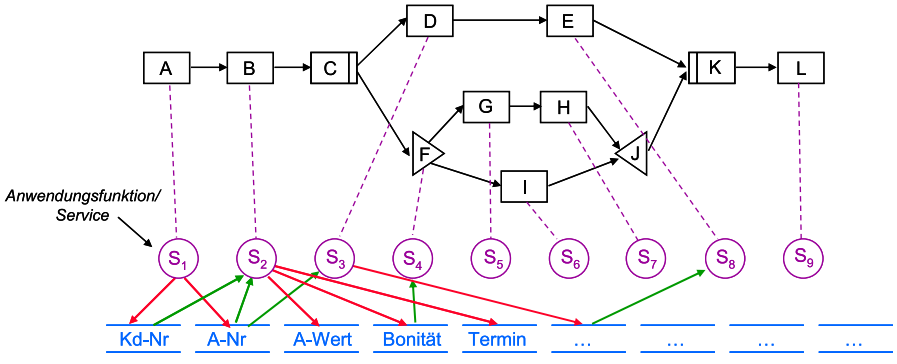


Abb. 5 Datenflüsse zwischen Anwendungsfunktionen

vorgeplanten Ablauf abweichen kann, also das Prozessmodell dieser Instanz durch Einfügen, Ändern, Löschen oder Verschieben von Prozessschritten individuell verändern kann. Das denkbare Spektrum an Änderungen reicht hierbei von minimalen Eingriffen bis zur völligen Umgestaltung des weiteren Ablaufs der Prozessinstanz. Auf den ersten Blick ist die Vorstellung, dass man vom geplanten Prozessablauf individuell (und womöglich sogar noch massiv) abweichen kann, vielleicht erschreckend. Man führt in vielen Fällen ein PMS ja genau aus dem Grunde ein, weil die Mitarbeiter sich nicht strikt genug and den vorgesehenen Ablauf halten. Dass ein PMS evtl. die technischen Möglichkeiten bietet, vom vorgeplanten Prozessablauf abzuweichen, bedeutet jedoch noch lange nicht, dass dies nun jedermann tun kann. Das heißt, ob und wer von dieser Möglichkeit ggf. Gebrauch machen darf, muss natürlich durch geeignete Autorisierungskonzepte festgelegt werden [17].

Bietet ein PMS keinerlei Flexibilität zur Ausführungszeit an, dann bedeutet dies, dass man entweder, egal was passiert, stur am vorgegebenen Prozessablauf festhalten muss (was in der Realität oftmals überhaupt nicht möglich ist) oder dass man das Problem unter Umgehung oder durch „Austricksen“ des PMS lösen muss, etwa durch Eingriffe in dessen „Innereien“ mit manuellem Ändern von Prozesszustandsdaten. Wenn man sich nur einmal vor Augen hält, was einen selbst bei den einfachen Dingen im täglichen und geschäftlichen Leben bereits gezwungen hat, vom geplanten Ablauf abzuweichen, dann wird einem rasch klar, dass sich eigentlich nicht die Frage stellt, wo man eventuell Flexibilität benötigt, sondern vielmehr, wo man diese Notwendigkeit von vornherein mit Sicherheit ausschließen kann. Man kann eben nicht alle Eventualitäten des Lebens vorplanen. Das heißt, man muss in der gegebenen Situation einen Weg finden, um das Problem zu lösen bzw. die zu erledigende Aufgabe dennoch zu bewältigen.

Darüber hinaus gibt es natürlich sehr viele Anwendungen, bei denen der Bedarf nach Flexibilität zur Laufzeit offensichtlich ist. Beispielhaft seien hier industrielle Entwicklungsprozesse von komplexen Produkten, wie etwa einem Fahrzeug oder einem Flugzeug, genannt, bei denen oftmals hunderte oder sogar tausende von Komponenten auf einander abgestimmt entwickelt und getestet werden müssen. Wenn sich hier während des Entwicklungsprozesses herausstellt, dass es Interferenzen zwischen Komponenten gibt, dann muss man sehr gezielt und problemspezifisch vorgehen, um die Ursachen zu lokalisieren, die notwendigen Änderungen anzuwenden und anschließend alle damit in Zusammenhang stehende Tests nochmals gezielt durchführen [10, 16]. Oder nehmen wir Behandlungsprozesse im Krankenhaus. Da jeder Patient anders ist, kann jegliche Art von vorgeplantem Ablauf stets nur ein Vorschlag sein, der von den behandelnden Ärzten immer daraufhin zu überprüfen ist, ob dieser im konkreten Einzelfall anwendbar bzw. angemessen ist. Wenn sich während der Therapie z. B. herausstellt, dass eine andere Erkrankung als die ursprünglich diagnostizierte vorliegt, muss möglicherweise der Behandlungsprozess ab diesem Zeitpunkt komplett geändert werden. Kein Prozessmodellierer kann den Ärzten diese Entscheidung und Verantwortung abnehmen [2, 8]. Bei beiden Beispielen ist es auch wichtig, im Einzelfall später einmal nachvollziehen zu können, wie der konkrete Ablauf seinerzeit war.

Der Vorrat an Beispielen aus verschiedensten Anwendungsdomänen, in denen eine systemseitige Prozessunterstützung wünschenswert ist, aber

starre Prozesse nicht in Frage kommen, ist praktisch unerschöpflich. Entsprechend groß ist das Anwendungspotenzial für PMS, die Flexibilität zur Ausführungszeit in geeigneter Weise unterstützen können. Ein PMS, welches in ausreichendem Maße Flexibilität zur Laufzeit anbieten kann, eröffnet darüber hinaus eine Fülle neuer Einsatzmöglichkeiten, die wir hier aus Platzgründen nur anreißen können:

- Verwaltung von Sonderfällen und vorgenommenen Anpassungen in einer Wissensbasis. Das Prozessmodell repräsentiert dann nur noch den Normalfall; dynamische Änderungen der Prozessinstanz können bei Bedarf, ggf. unter Wiederverwendung einer früher getroffenen Entscheidung aus der Wissensbasis, flexibel vorgenommen werden [18].
- Sehr viel rascheres Einsetzen von Prozessen, da man eventuell bestehende Schwächen im Prozessmodell auch noch im laufenden Betrieb durch Änderung der Prozessinstanz „ausbügeln" kann.
- Inkrementelle Fortschreibung von Prozessen: Diese ist für lange laufende Prozesse (z. B. bei lang laufenden Bauvorhaben) interessant, bei denen man die späteren Phasen noch nicht im letzten Detail festlegen will. Mittels Änderung auf Prozessinstanzebene kann man solche Prozesse nach und nach fortschreiben.
- Individuelle Prozesse: Im Prinzip kann man eine Prozessinstanz beim Start auch mit einem „privaten" individuellen Prozessmodell ausstatten, das spezifisch für den konkreten Einzelfall (z. B. von einem Entscheidungsunterstützungssystem) generiert wurde [1].

Wie sieht es nun bezüglich Flexibilität zur Ausführungszeit bei den derzeit angebotenen PMS aus? Hier muss man leider feststellen, dass die Hersteller von allgemein einsetzbaren PMS die Anforderung nach Flexibilität zur Ausführungszeit bis vor kurzem praktisch völlig ignoriert haben. Erst seit relativ kurzer Zeit bieten einige PMS die Möglichkeit, zumindest einfache Ad-hoc-Abweichungen, wie etwa das Ersetzen eines Prozessschrittes durch einen anderen Prozessschritt mit kompatibler Schnittstelle (im Sinne von „late binding") oder aber das einfache Einfügen eines Prozessschrittes an der aktuellen Position, durchführen zu können. Da viele PMS, wie bereits erwähnt, die zwischen den Anwendungsfunktionen stattfindenden Datenflüsse nicht kennen, können aber selbst diese einfachen Maßnahmen kritisch für die Robustheit der weiteren Prozessausführung sein, sodass solche Änderungen auf Prozessinstanzebene eigentlich nur von Prozessspezialisten vorgenommen werden können. Größere bzw. komplexere Änderungen beherrschen die gängigen PMS derzeit bei weitem nicht.

Prozessschemaevolution

Prozessmodelle sind ihrer Natur nach immer wieder Änderungen unterworfen, sei es, dass ein Unternehmen intern umstrukturiert wird, dass Aufgaben ausgelagert oder ins Unternehmen geholt werden, dass das alte Prozessmodell Schwächen aufweist oder dass geänderte gesetzliche oder andere Rahmenbedingungen eine Prozessänderung erforderlich machen. Wird von einem Prozessmodell A (auch *Prozessschema* genannt) eine neue Version A' abgeleitet, so folgen alle auf Basis von A gestarteten Prozessinstanzen erst einmal weiterhin dem in A hinterlegten Prozessmodell. Gibt es jedoch schwerwiegende Probleme mit dem bisherigen Prozess oder muss man ab einem gewissen Stichtag neue Vorgaben erfüllen, dann muss man die auf dem alten Prozessschema bereits gestarteten Prozessinstanzen irgendwie auf die neue Verfahrensweise umstellen.

Verfügt das PMS über die Fähigkeit zur Abweichung auf Prozessinstanzebene, könnte man im Prinzip dieses Mittel einsetzen, um das neue Verhalten zu erzielen. Sind jedoch bereits sehr viele Prozessinstanzen dieses Typs unterwegs, ist diese Vorgehensweise meist zu aufwendig und fehlerträchtig. Sehr viel eleganter ist in einem solchen Fall die *Prozessschemaevolution*. Hierunter versteht man die Fähigkeit eines PMS, Prozessinstanzen auf ein geändertes Schema bzw. Prozessmodell migrieren zu können. Als Minimalforderung wird man erwarten, dass dies zumindest für die unveränderten Prozessinstanzen angeboten wird, aber natürlich sollte es diese Möglichkeit auch für individuell geänderte Prozessinstanzen geben. Natürlich kann man nicht jede Prozessinstanz so einfach auf das neue Schema „umhängen". So ergibt es z. B. keinen Sinn (und kann sogar in der Folge zu Laufzeitfehlern führen), Prozessinstanzen zu migrieren, die im Ablauf bereits zu weit fortgeschritten sind oder bei denen die Schemaänderung mit individuellen Instanzänderungen in Konflikt steht [14].

Die kommerziellen Hersteller von PMS haben mittlerweile erkannt, dass sie hier etwas anbie-

ten müssen, und so sieht man inzwischen einige Systeme, die diese Fähigkeit aufweisen. In Anbetracht der Tatsache, dass die wenigsten Hersteller bei eventuellen Korrektheitsanalysen die Datenflüsse mit einbeziehen, sollte der Prozessmodellierer bzw. -verantwortliche jedoch sehr genau prüfen, ob alle vom PMS ggf. zur Migration angebotenen Instanzen auch tatsächlich ohne Konsistenzprobleme migrierbar sind. Andernfalls drohen Inkonsistenzen oder Laufzeitfehler. Nachdem von den gängigen PMS, wenn überhaupt, nur rudimentäre Unterstützung von Abweichungen auf Prozessinstanzebene angeboten werden, wird die Menge der PMS, welche eine Migration von individuell geänderten Prozessinstanzen unterstützen, derzeit sehr klein (wenn nicht sogar leer) sein.

Durchgängige Prozessflexibilität am Beispiel von ADEPT

Im Folgenden soll am Beispiel der von uns entwickelten ADEPT-Technologie bzw. der darauf basierenden AristaFlow® BPM Suite [21] gezeigt werden, dass man PMS (und darauf aufbauend prozessorientierte Informationssysteme) realisieren kann, welche die obigen Anforderungen an Flexibilität sowohl zur Entwurfs- als auch zur Ausführungszeit bereits weitgehend erfüllen können. Auslöser dieser Forschungsarbeiten war ein umfangreiches Forschungsprojekt im klinischen Bereich, das von 1992 bis 1994 in Zusammenarbeit mit verschiedenen Partnern des Universitätsklinikums Ulm durchgeführt wurde und das die Erarbeitung von Konzepten für klinische Informationssysteme der nächsten Generation zum Gegenstand hatte [5, 6]. In diesem Zusammenhang haben wir uns u. a. sehr intensiv mit medizinisch-organisatorischen Abläufen und deren Anforderungen befasst. Dabei wurde mehr und mehr klar, dass die Einführung *prozessorientierter* klinischer Informationssysteme für das Klinikpersonal einen wirklich großen Fortschritt mit sich bringen würde.

Allerdings war auch klar, dass man eine Reihe konfliktärer Ziele unter einen Hut bringen muss. So sollte z. B. das Prozessmodell ausdrucksstark genug sein, um alle im klinischen Umfeld auftretenden Prozessstrukturen geeignet modellieren zu können. Das heißt, es müssen nicht nur alternative Pfade, sondern auch Parallelpfade und zyklische Ausführungen abgebildet werden können. Außerdem kann man sich nicht auf formular- oder dokumentenorientierte Prozesse beschränken, sondern es müssen Anwendungssysteme bzw. -dienste jeglicher Art integrierbar sein, etwa zur Ansteuerung von Laborautomaten oder für den Zugriff auf Terminplanungssysteme. Des Weiteren müssen in der klinischen Anwendungsdomäne neue Prozesse rasch einführbar und bestehende Prozesse rasch änderbar sein. Dies wiederum bedeutet, dass die Komplexität für die Anwendungsentwickler nicht signifikant ansteigen darf; und dies gelingt nur, wenn Modellierungsfehler der vorangehend beschriebenen Art weitgehend systemseitig erkannt oder – noch besser – von vornherein ausgeschlossen werden. Außerdem, und das war mit die größte sich uns stellende Herausforderung, können Prozesse in dieser Domäne nicht starr sein. Flexibilität zur Ausführungszeit ist hier ein absolutes Muss! Allerdings nützt Flexibilität zur Ausführungszeit nichts, wenn diese in der Handhabung „vor Ort“ so kompliziert ist, dass man hierfür stets einen Prozessspezialisten benötigt. Das heißt, eine ganz wesentliche Herausforderung war, die ganze damit einhergehende Komplexität vor den Anwendern, aber letztlich auch vor den Prozessmodellierern und Entwicklern von Anwendungsfunktionen, unter der Systemoberfläche zu verstecken und „nach oben“ einfach zu bedienende Schnittstellen anzubieten. Ausführlichere Darstellungen der Entwicklungsgeschichte des ADEPT-Projekts sowie der technologischen Lösungen finden sich in [4], auf den Seiten des AristaFlow-Forums [22] sowie den Webseiten des Instituts für Datenbanken und Informationssysteme der Universität Ulm [23].

Prozessmodellierung mittels „Correctness by Construction“

Ein wesentliches Anliegen beim Entwurf des ADEPT-Prozess-Meta-Modells (Abb. 6) war, dass es ausdrucksstark genug sein sollte, um alle wichtigen Prozessstrukturen geeignet abbilden zu können. Als „Messlatte“ dienten zahlreiche Prozessbeispiele aus einem weiteren Forschungsprojekt [11]. Das Prozess-Meta-Modell sollte aber auch eingeschränkt genug sein, um möglichst viele Modellierungsfehler durch formale Analysen erkennen zu können bzw. sie von vornherein auszuschließen. Das unter diesen Randbedingungen entwickelte Prozess-Meta-Modell ist trotz seiner Blockstruktur aufgrund verschiedener Erweiterungen sehr ausdrucksstark, hat eine solide formale Basis und erlaubt effiziente Überprüfungen

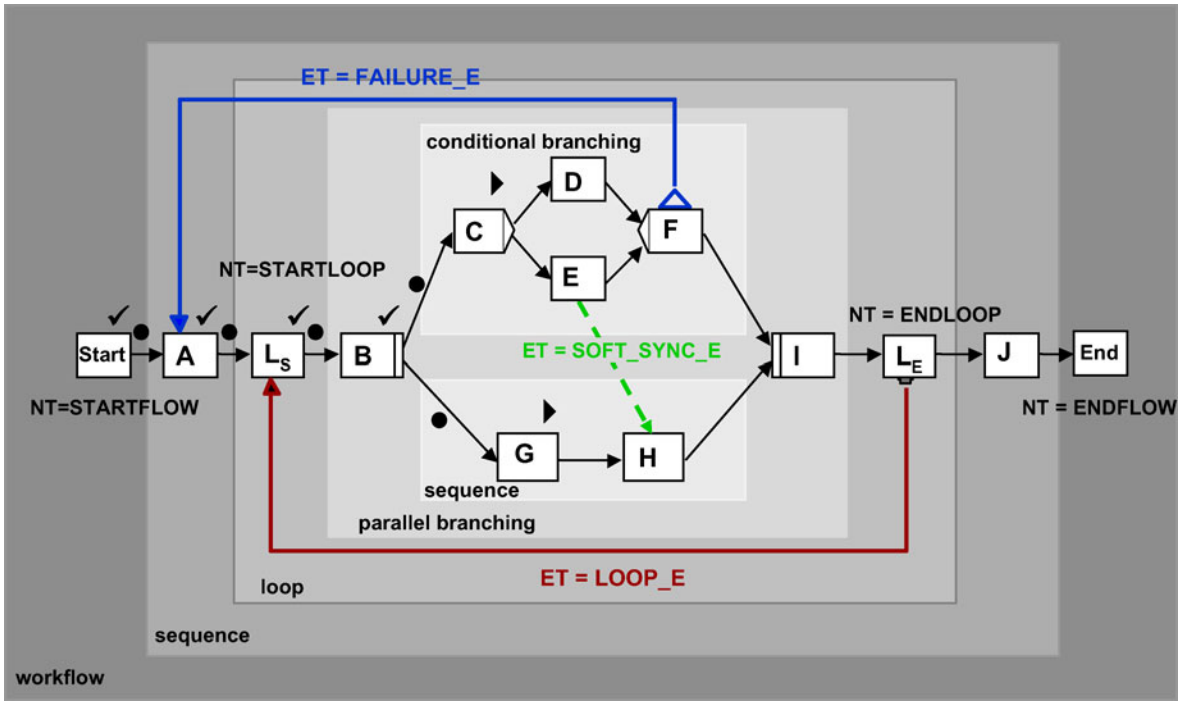


Abb. 6 ADEPT Prozess-Meta-Modell

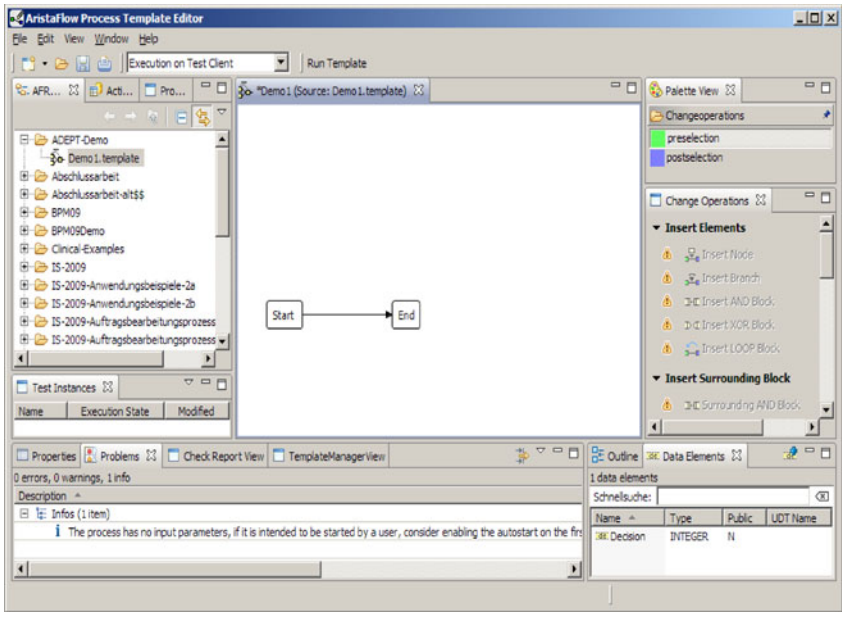


Abb. 7 AristaFlow Process Template Editor

der erstellten Prozessmodelle auf Korrektheit. Im Kontext von Ad-hoc-Abweichungen (s. u.) ermöglicht es darüber hinaus, sehr rasch zu entscheiden, ob und – falls ja – wie eine gewünschte Änderung realisiert werden kann [3, 12, 13].

Im Gegensatz zur gängigen Praxis, das Prozessmodell erst einmal relativ freihändig komponieren zu lassen und es dann anschließend auf Fehler zu analysieren, wird der Prozessmodellierer in ADEPT bzw. AristaFlow von Anfang an so geführt, dass viele Modellierungsfehler „per Konstruktion" erst gar nicht gemacht werden können. Wenn man z. B. ein neues Prozessmodell erstellt, startet der *AristaFlow Process Template Editor* mit einem in-

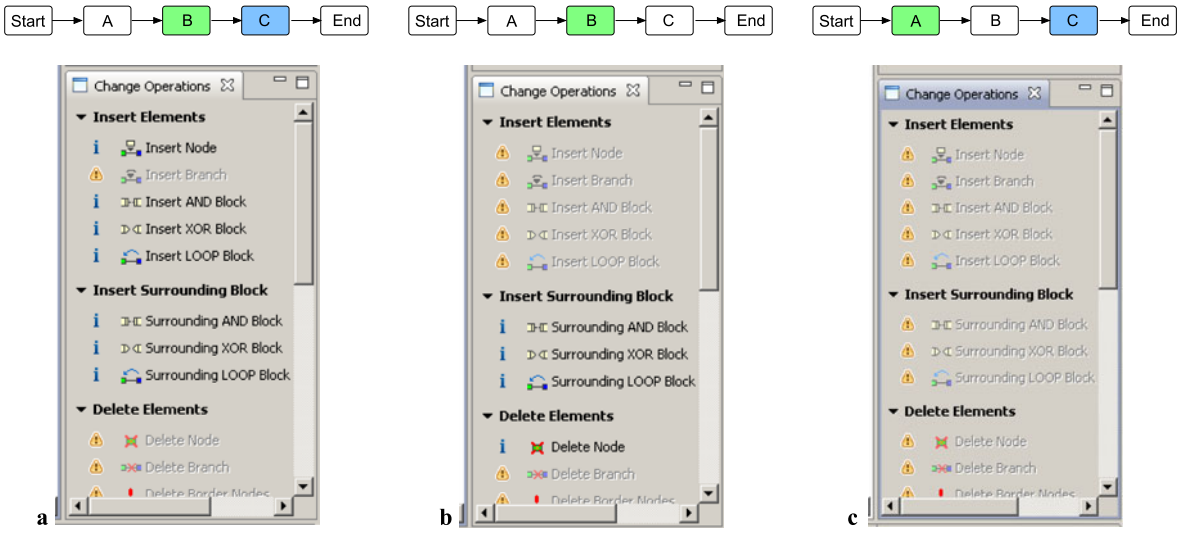


Abb. 8 Markierungen im Prozessmodell und freigeschaltete Operationen

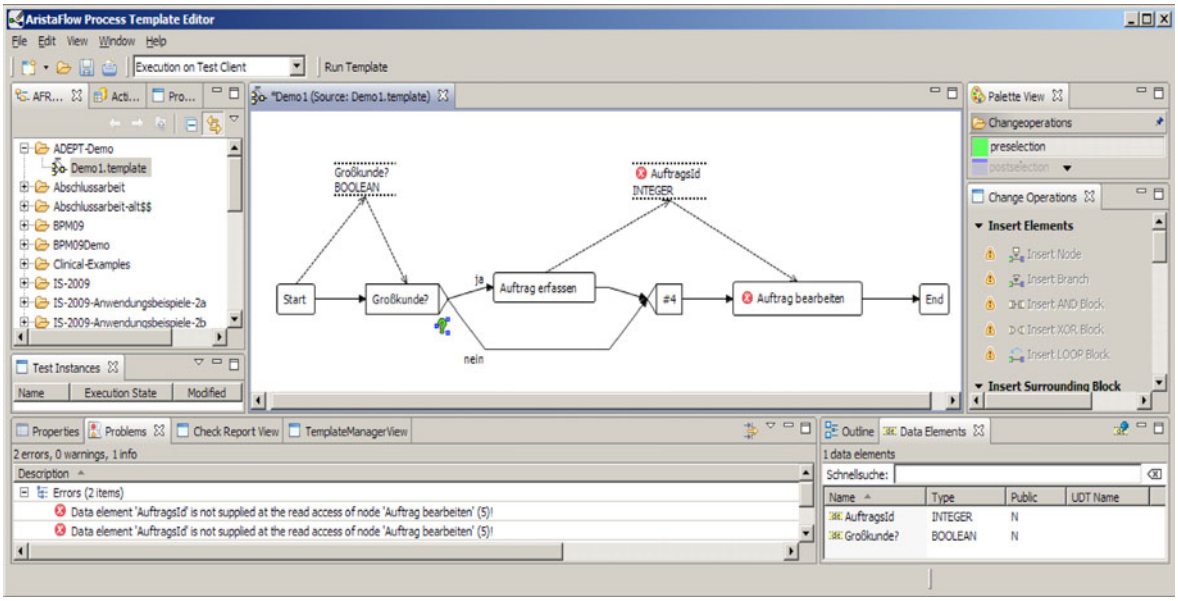
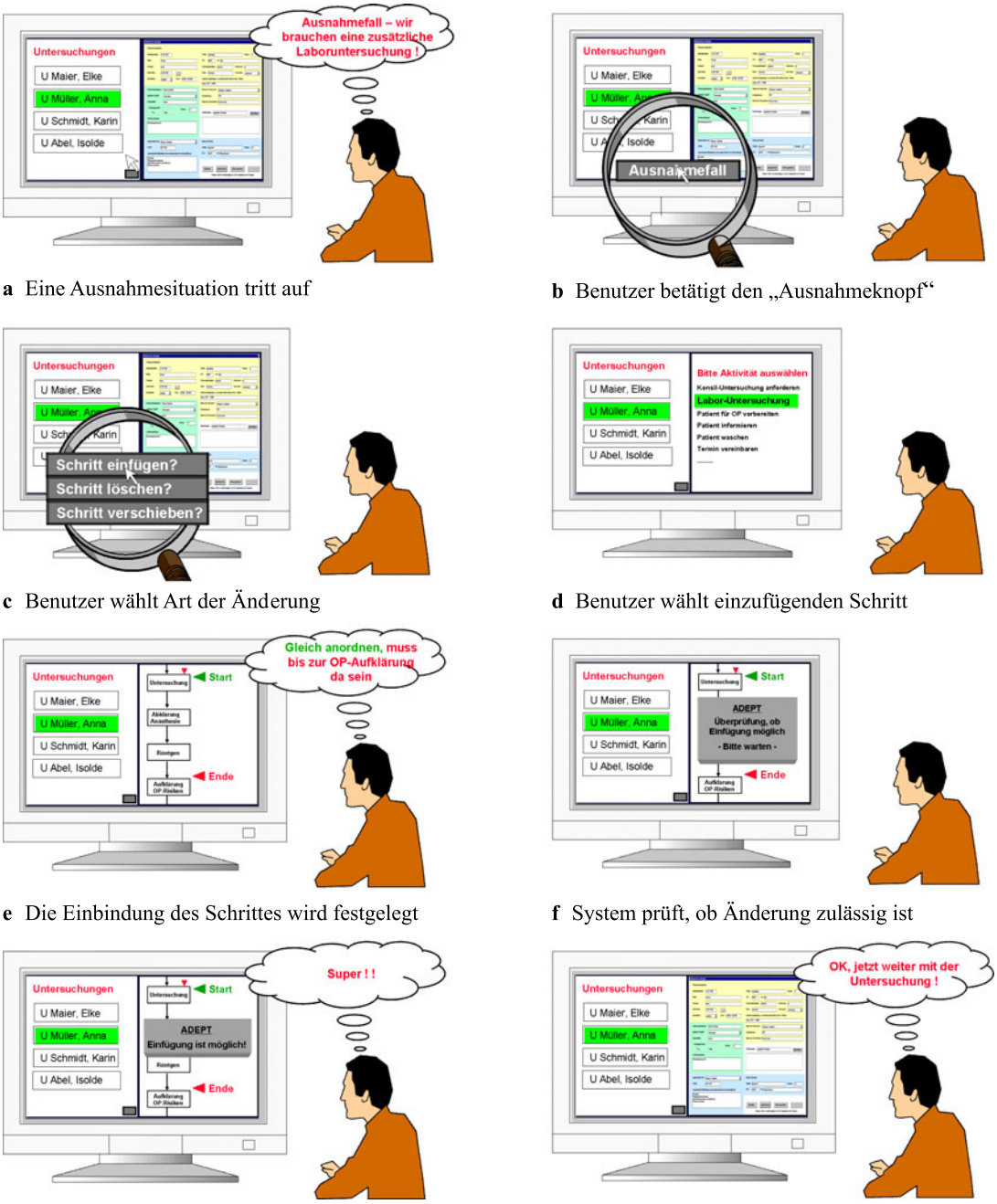


Abb. 9 Erkennung eines inkonsistenten Datenflusses

itialen Prozessmodell, das nur aus einem Start- und einem End-Knoten besteht und welches das kleinstmögliche, strukturell konsistenteste Prozessmodell darstellt (Abb. 7). Im Teilfenster „Change Operations“ werden jeweils die Operationen angezeigt und freigeschaltet, die auf das angezeigte Prozessmodell und die dort ggf. getroffene Auswahl anwendbar sind. Da im angezeigten Prozessmodell nichts ausgewählt bzw. markiert wurde, sind rechts keine Operationen wählbar (mit Ausnahme von „Insert Data Element“ das in der Liste weiter unten kommt und deshalb in Abb. 7 nicht angezeigt wird).

Mittels Preselection (= grün bzw. hellgrau in einem Graustufenausdruck) und Postselection (= blau bzw. dunkelgrau in einem Graustufenausdruck) kann ein Bereich im Prozessmodell markiert werden, woraufhin die dann wählbaren Operationen freigeschaltet werden. Abbildung 8 zeigt exemplarisch das Zusammenspiel zwischen Markierungen im Prozessmodell und Freischaltung von Änderungsoperationen. Alle freigeschalte-



a Eine Ausnahmesituation tritt auf

b Benutzer betätigt den „Ausnahmeknopf“

c Benutzer wählt Art der Änderung

d Benutzer wählt einzufügenden Schritt

e Die Einbindung des Schrittes wird festgelegt

f System prüft, ob Änderung zulässig ist

g Die Änderung kann durchgeführt werden

h Benutzer setzt Tätigkeit fort

Abb. 10 Durchführung einer Ad-hoc-Änderung durch Endbenutzer

ten Operationen sind in ihrer Wirkung auf das Prozessmodell eindeutig bestimmt und überführen es aus einem strukturell konsistenten Zustand in einen neuen strukturell konsistenten Zustand. Andere Inkonsistenzen, die nicht schon auf diese Weise „per Konstruktion“ vermieden werden können, wie etwa unversorgte Aufrufparameter von Prozessschritten (bzw. den diesen Schritten zugeordneten Anwendungsfunktionen), werden durch nebenläufige Analysen erkannt und im „Problems“-Fenster anzeigt. In dem in Abb. 9 dargestellten Prozess wird z. B. das Datenelement „AuftragsId“

vom Schritt „Auftrag bearbeiten“ gelesen, d. h. der Inhalt dieses Datenelements wird von diesem Prozessschritt zur Ausführungszeit als „Input“ benötigt. Im dargestellten Prozessmodell wird dieses Datenelement jedoch nur im oberen Zweig der XOR-Verzweigung „versorgt“. Dieser Fehler im Datenfluss wird sofort erkannt und durch Markierung der betroffenen Elemente sowie als Text im „Problems“-Fenster (siehe unteres Teilfenster in Abb. 10) angezeigt.

ADEPT wurde dafür konzipiert, dass sich Anwendungsfunktionen verschiedenster Art integrieren lassen. Dazu werden diese dem Prozessmodellierer in homogener Weise angeboten, d. h. von internen Implementierungs- und Aufrufdetails wird abstrahiert. Nach Registrierung einer Anwendungsfunktion im sog. *Activity Repository* steht diese im *Process Template Editor* zur Verfügung und kann dann einfach aus dem entsprechenden Teilfenster mittels Drag and Drop auf einen Knoten oder eine Kante im Prozessmodell gezogen werden. In diesem Zusammenhang prüft das System, ob für die Anbindung eventueller Ein- und Ausgabeparameter dieser Anwendungsfunktion bereits typgerechte Datenelemente im Prozessmodell vorliegen und bietet diese ggf. zur Verknüpfung an. Hierbei wird der resultierende Datenfluss wieder auf Korrektheit geprüft. Die oben skizzierte Führung der Anwender bei der Modellierung, in Verbindung mit umfassenden Datenflussanalysen sowie weiteren nebenläufig durchgeführten Überprüfungen, bezeichnen wir als „*Correctness by Construction*“. Sie hilft die Entwicklungs- und Testzeit für Prozesse signifikant zu reduzieren.

Ad-hoc-Abweichungen zur Ausführungszeit

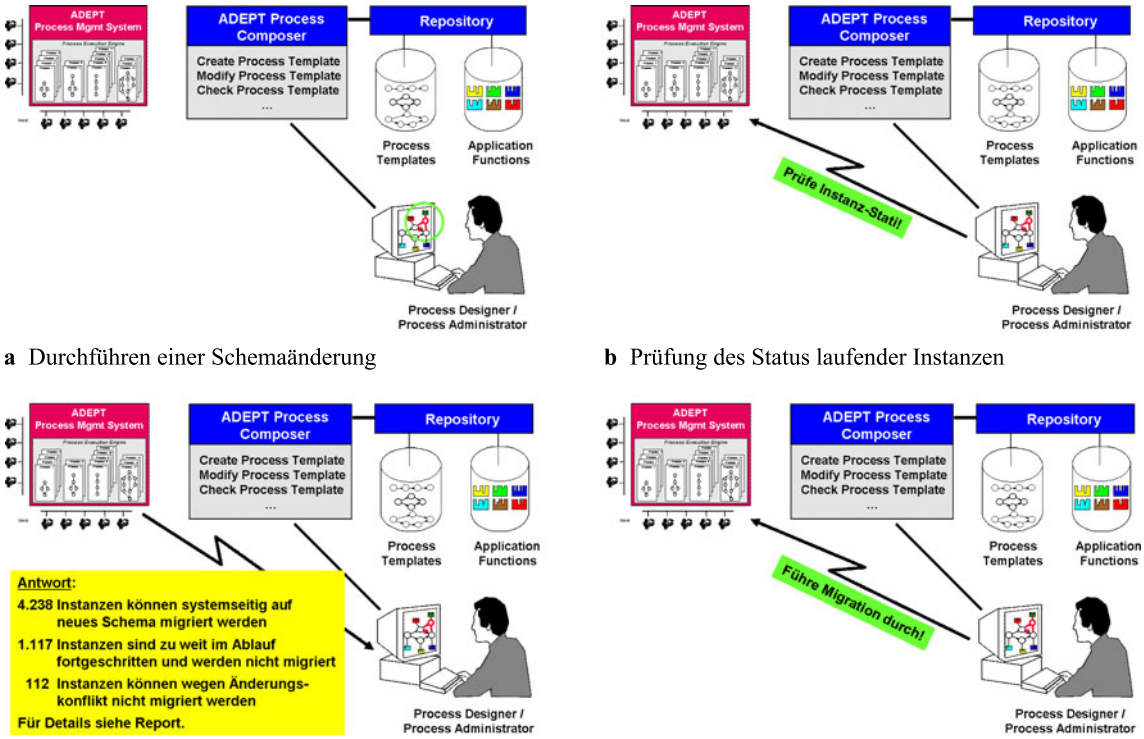
In Bezug auf die Unterstützung von Ad-hoc-Änderungen bestanden mehrere Herausforderungen. Erstens war das volle Spektrum an Änderungsoperationen anzubieten, zweitens sollten dabei keine Abstriche in Bezug auf Konsistenzüberprüfungen gemacht werden und drittens waren die Schnittstellen so zu gestalten, dass – zumindest in einfach gelagerten Fällen – Endbenutzer mit entsprechender Autorisierung die Ad-hoc-Änderungen auf Prozessinstanzebene selbstständig durchführen können. Unser Lösungsansatz ist es, semantisch hohe Schnittstellen für Ad-hoc-Änderungen zu realisieren und die damit inhärent verbundene Komplexität unter der Systemoberfläche zu verstecken. Die auf Systemebene zu bedienende (Service-) Schnittstelle benötigt z. B. für das Einfügen eines Prozessschrittes im Wesentlichen nur die folgenden Informationen:

1. Um welche Anwendungsfunktion handelt es sich?
2. Nach welchem Prozessschritt soll der neue Prozessschritt zur Ausführung angeboten werden?
3. Vor welchem Prozessschritt soll der einzufügende Schritt abgeschlossen sein?

Abbildung 9 illustriert, wie man auf Basis dieser Systemschnittstelle die Interaktion mit dem Anwender realisieren könnte. In diesem Beispiel soll ein Oberarzt die Entscheidung eines Ambulanzarztes hinsichtlich eines operativen Eingriffs absegnen bzw. ablehnen. Im konkreten Fall möchte er jedoch selbst noch eine Laboruntersuchung veranlassen, was an sich im Prozessmodell nicht vorgesehen ist. Da er zu den autorisierten Personen gehört, die in eingeschränktem Umfang Ad-hoc-Abweichungen vornehmen dürfen, kann er via „Ausnahmeknopf“ und den dort angebotenen Operationen gewisse Änderungen an der Prozessinstanz vornehmen. Die vorgenommenen Änderungen werden im Prozesslog protokolliert und können deshalb später jederzeit im Detail nachvollzogen werden. Dem Prozessspezialisten steht natürlich eine mächtigere Schnittstelle zur Durchführung von Ad-hoc-Änderungen zur Verfügung. Diese bietet praktisch die volle funktionale Mächtigkeit des *Process Template Editors*, wiederum unter (hier erweiterter) Beachtung des „Correctness by Construction“-Prinzips.

Prozessschemaevolution

Im ADEPT-Projekt wurden grundlegende und umfassende Konzepte für Prozessschemaevolution entwickelt. Diese ermöglichen nicht nur die Migration unveränderter, sondern auch individuell veränderter Prozessinstanzen. Bei der systemseitigen Überprüfung wird nicht nur berücksichtigt, ob die jeweils betrachtete Prozessinstanz in ihrem Ausführungszustand zu weit fortgeschritten ist, sondern es wird auch geprüft, ob eventuell vorgenommene individuelle (Ad-hoc-)Änderungen mit den Änderungen auf Schemaebene harmonieren bzw. in Konflikt stehen [15]. Des Weiteren werden hier wieder die bereits erwähnten Konsistenzregeln beachtet. Leider können wir aus Platzgründen nicht im Detail auf dieses



a Durchführen einer Schemaänderung

b Prüfung des Status laufender Instanzen

c Ergebnis der Prüfung

d Durchführung von Instanz-Migrationen

Abb. 11 Prozessschemaevolution

mächtige (und weltweit in dieser Form bislang einmalige) Feature der ADEPT-Technologie eingehen. Stattdessen begnügen wir uns mit einer Illustration (Abb. 11), die zeigen soll, wie man sich den Ablauf einer Prozessschemaevolution vorzustellen hat.

Der Prozessmodellierer lädt das zu modifizierende Prozessmodell bzw. -schema, führt die gewünschten Änderungen durch und speichert eine neue Version dieses Prozessmodells (Abb. 11a). Anschließend kann er das ADEPT-System prüfen lassen, welche der auf dem alten Schema basierenden Prozessinstanzen auf das neue Schema migrierbar sind (Abb. 11b). Er erhält eine (in Abb. 11c stark vereinfacht dargestellte) Liste dieser Instanzen mit näheren Angaben, warum eine Instanz systemseitig als migrierbar oder nicht migrierbar eingestuft wurde. Aus der Liste der als migrierbar eingestuften Instanzen kann er über Filter bei Bedarf noch Instanzen abwählen, d. h. von der Migration ausschließen. Anschließend beauftragt er das System, die verbliebenen Instanzen auf das neue Schema zu migrieren (Abb. 11d). Wird zu einem späteren Zeitpunkt eine weitere Version von diesem Prozessmodell abgeleitet, wären alle noch laufenden Instanzen dieses Schemas (also auch die zuvor erst auf dieses Schema migrierten) wieder Kandidaten für eine sich evtl. erneut anschließende Prozessschemaevolution.

Fazit

Die Unterstützung betrieblicher Abläufe durch ein PMS bietet vielfältige Vorteile, etwa im Hinblick auf die aktive Koordination der Prozesse (z. B. Reduzierung von Leerlaufzeiten und falschen Bearbeitungsreihenfolgen), die Möglichkeit der (Teil-)Automatisierung der Prozesse oder die verbesserte Einhaltung von Compliance-Vorgaben einschließlich entsprechender Nachweise. Die bislang in den Unternehmen überwiegend manuell ausgeführten Prozesse haben viele Schwächen und Nachteile, besitzen aber auch eine große Stärke: In Ausnahmesituationen kann man sehr flexibel reagieren und zur Not auch einmal komplett vom vorgeplanten Ablauf abweichen. Die gängigen PMS besitzen diese Eigenschaft nicht bzw. nur in sehr eingeschränktem Maße. Damit sind ihre Einsatzmöglichkeiten stark eingeschränkt.

Am Beispiel der ADEPT- bzw. AristaFlow-Technologie haben wir gezeigt, dass man techno-

logisch durchaus in der Lage ist, PMS zu entwickeln, die um Größenordnungen flexibler sind als die bisherigen Systeme und diesen eigentlich in allen die Flexibilität betreffenden Belangen – d. h. der Flexibilität zur Entwurfszeit, der Flexibilität zur Ausführungszeit sowie der Prozessschemaevolution – haushoch überlegen sind. Voraussetzung hierfür ist, dass man sich bei der Entwicklung von PMS darauf zurückbesinnt, was die wirklich fundamentalen Anforderungen an diese Systeme sind und hierfür die geeigneten technologischen Konzepte entwickelt. Es ist unschwer vorauszusagen, dass den PMS nur dann ein ähnlicher Siegeszug wie den relationalen Datenbanksystemen beschert sein wird, wenn es gelingt, sie ebenfalls auf eine sehr viel mächtigere Technologiebasis zu stellen, die es zudem ermöglicht, die komplizierten Dinge weitgehend „unter der Oberfläche" zu erledigen und „nach oben" einfach zu bedienende Schnittstellen anzubieten. Gelingt dies nicht, werden PMS eine Nischentechnologie mit eng begrenztem Einsatzbereich bleiben.

Literatur

1. Bassil S, Benyoucef M, Keller RK, Kropf P (2002) Addressing Dynamism in E-negotiations by Workflow Management Systems. Proc Int Workshops on Database and Expert Systems Applications, DEXA'2002, Third Int Workshop on Negotiations in e-Markets – Beyond Price Discovery, September 2002, Aix-en-Provence, France
2. Dadam P, Reichert M, Kuhn K (2000) Clinical Workflows – The Killer Application for Process-oriented Information Systems? Proc Int Conf on Business Information Systems, BIS 2000, 4th Int Conf., April 2000, Poznan, Poland, Springer, pp 36–59
3. Dadam P, Reichert M (2009) The ADEPT Project: A Decade of Research and Development for Robust and Flexible Process Support. Comput Sci Res Dev 23(2):81–98
4. Dadam P, Reichert M, Rinderle-Ma S, Göser K, Kreher U, Jurisch M (2009) Von ADEPT zur AristaFlow BPM Suite – Eine Vision wird Realität: „Correctness by Construction" und flexible, robuste Ausführung von Unternehmensprozessen. EMISA FORUM 1:9–28
5. Kuhn K, Reichert M, Nathe M, Beuter T, Dadam P (1994) An Infrastructure for Cooperation and Communication in an Advanced Clinical Information System. Proc. 18th Ann Sym on Computer Applications in Medical Care 1994, SCAMC '94, Washington, pp 519–523
6. Kuhn K, Reichert M, Nathe M, Beuter T, Heinlein C, Dadam P (1994) A Conceptual Approach to an Open Hospital Information System. Proc 12th Int Congress on Medical Informatics, MIE '94, May 1994, Lissabon, pp 374–378
7. Laue R, Mendling J (2008) The Impact of Structuredness on Error Probability of Process Models. Proc. 2nd Int United Information Systems Conf, UNISCON 2008, April 2008, Klagenfurt, Österreich, LNBIP 5, pp 585–590
8. Lenz R, Reichert M (2007) IT Support for Healthcare Processes – Premises, Challenges, Perspectives. Data Knowl Eng 61(1):39–58
9. Mendling J (2009) Empirical Studies in Process Model Verification. Transactions on Petri Nets and Other Models of Concurrency II. Special Issue on Concurrency in Process-Aware Information Systems, LNCS 5460, pp 208–224
10. Müller D, Reichert M, Herbst J (2007) Data-driven Modeling and Coordination of Large Process Structures. Proc. Int Conf on Cooperative Information Systems, CoopIS, November 2007, Vilamoura, Algarve, Portugal, LNCS 4803, pp 131–149
11. Reichert M, Dadam P, Mangold R, Kreienberg R (2000) Computerbasierte Unterstützung von Arbeitsabläufen im Krankenhaus – Konzepte, Technologien und deren Anwendung. Zentralbl Gynäkol 1:53–67
12. Reichert M, Dadam P (1998) $ADEPT_{flex}$ – Supporting Dynamic Changes of Workflows Without Losing Control. J Intell Inf Syst 10(2):93–129
13. Reichert M (2000) Dynamische Ablaufänderungen in Workflow-Management-Systemen. Dissertation, Universität Ulm, Fakultät für Informatik
14. Rinderle S, Reichert M, Dadam P (2004) Correctness Criteria for Dynamic Changes in Workflow Systems – a Survey. Data Knowl Eng 50(1):9–34
15. Rinderle SB (2004) Schema Evolution in Process Management Systems. Dissertation, Universität Ulm, Fakultät für Informatik
16. Vanderfeesten I, Reijers HA, Aalst WMP (2008) Product Based Workflow Support: Dynamic Workflow Execution. Proc Int Conf on Advanced Information Systems Engineering, CAiSE 2008, Juni 2008, Montpellier, France, LNCS 5074, pp 571–574
17. Weber B, Reichert M, Wild W, Rinderle S (2005) Balancing Flexibility and Security in Adaptive Process Management Systems. Proc. 13th Int Conf on Cooperative Information Systems (CoopIS '05), Agia Napa, Cyprus, LNCS 3760, pp 59–76
18. Weber B, Reichert M, Rinderle-Ma S, Wild W (2009) Providing Integrated Life Cycle Support in Process-Aware Information Systems. Int J Coop Inf Sys 18(1):115–165
19. Weske M (2007) Business Process Management. Springer, Berlin
20. http://www.workflowpatterns.com/patterns/control/, letzter Zugriff 30.6.2010
21. www.AristaFlow.com, letzter Zugriff 30.6.2010
22. www.AristaFlow-Forum.de, letzter Zugriff 30.6.2010
23. www.uni-ulm.de/dbis, letzter Zugriff 30.6.2010