Dieter Spath, Klaus Haasis, Dieter Klumpp (Hrsg.): Aktuelle Trends in der Softwareforschung - Tagungsband zum doIT-Forschungstag, Juni 2005, Karlsruhe, Juni 2005, S. 47-67

Auf dem Weg zu prozessorientierten Informationssystemen der nächsten Generation – Herausforderungen und Lösungskonzepte ♣

Peter Dadam[1] Manfred Reichert[2] Stefanie Rinderle[1] Colin Atkinson[3]

[1]Universität Ulm
Abt. Datenbanken und IS
Ulm

[2]University of Twente
Information Systems Group
Enschede, Niederlande

[3]Universität Mannheim
Lehrstuhl für Softwaretechnik
Mannheim

Zusammenfassung

Prozess-Management-Systeme müssen gegenüber dem heutigen Stand erheblich leistungsfähiger werden, damit prozessorientierte Informationssysteme für ein breites Spektrum von Anwendungen einsetzbar werden. Besonders wichtig ist in diesem Zusammenhang ist die Realisierung einer wesentlich höheren Flexibilität und Adaptivität als sie von heutigen Systemen geboten werden. In diesem Beitrag wird zunächst aus Anwendersicht aufgezeigt, wie der Umgang mit solchen flexiblen und adaptiven Prozess-Management-Systemen aus Anwendersicht aussehen könnte und was die daraus resultierenden technologischen Herausforderungen sind. Die vorgestellten Beispiele und Lösungsansätze orientieren sich an den im ADEPT-Projekt gewonnenen Erkenntnissen sowie den im Nachfolgeprojekt AristaFlow verfolgten Zielen.

1 Einleitung

"Prozessorientierung" und damit im Zusammenhang stehende Schlagworte und Trends dominieren schon seit einigen Jahren die Fachpresse. Man könnte daher vermuten, dass der Markt für Softwarelösungen zur Realisierung prozessorientierter Informationssysteme boomt. Dem ist aber nicht ganz so. Die Gründe hierfür sind vielfältig: Die Realisierung prozessorientierter Informationssysteme erfordert zum einen ein erhebliches Umdenken in den betroffenen Fachabteilungen, zum andern verursachen die Erfassung und Modellierung der (Geschäfts-) Prozesse erheblichen Zeitaufwand und Kosten. Ein weiterer Grund ist, dass auf dem Markt eine Vielzahl verschiedener Produkte und Technologien miteinander konkurrieren, die auf sehr unterschiedliche Weise versuchen, prozessorientierte IT-Lösungen zu realisieren. Der kleinste gemeinsame Nenner dieser Ansätze besteht darin, dass bei einem prozessorientierten Informationssystem eine Trennung von Prozesslogik und Applikationsfunktionen angenommen bzw. realisiert wird (siehe Abb. 1). Hinsichtlich der konkreten technischen Realisierung gibt es jedoch gravierende Unterschiede.

Lösungsangebote, die aus dem Bereich der *Anwendungsintegration* (*EAI – Enterprise Application Integration*) kommen, benutzen meist eine kommunikationsbasierte Infrastruktur (z.B. Message Broker), um Anwendungssysteme miteinander zu verbinden. Die graphische Modellierung des "Prozesses" wird hier im Wesentlichen dazu benutzt, die Nachrichten- bzw. Datenflüsse zwischen den Anwendungssystemen graphisch darzustellen sowie hieraus den erforderlichen Programmcode für den Nachrichtenaustausch und die zugehörenden Aktionen zu generieren. Ein Anwendungssystem A kann dann bei einer Änderung seiner Daten eine Nachricht eines bestimmten Typs auf den Nachrichtenbus ("Message Broker") legen und nachgelagerte Anwendungen B und C können diese Nachricht dann zur Kenntnis nehmen (um ggf. dann bei sich selbst nachgeschaltete Aktionen durchzuführen). Die Anwendungen B und C erzeugen dann ggf. ihrerseits wieder Nachrichten, die sie auf den Nachrichtenbus legen. Die Nachrichten müssen alle erforderlichen Informationen ("Aufrufparameter") enthalten, damit auf der Empfängerseite jeweils der konkrete Kontext sowie die auszuführenden Operationen genau bestimmt werden können. – Ein Abweichen vom vorgeplanten Ablauf ist bei diesen Systemen üblicherweise nicht möglich, da die Ablauflogik hart im generierten Programmcode verdrahtet ist.

♣ Diese Arbeit wurde vom Land Baden-Württemberg im Rahmen des AristaFlow-Projektes [1] gefördert.

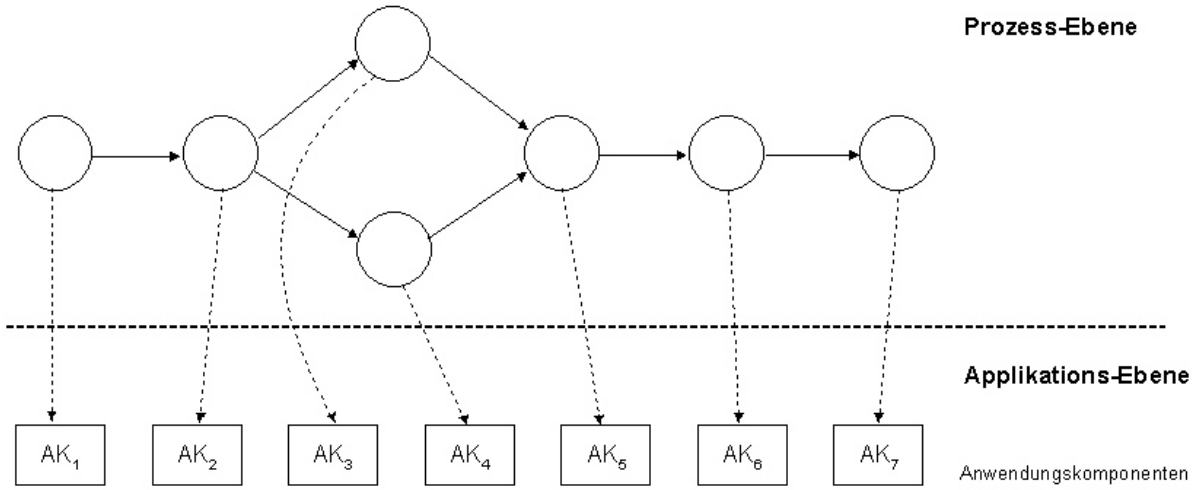


Abb. 1: Trennung von Prozesslogik und Anwendungsfunktionen

Lösungsangebote, die aus dem *Verwaltungs- und Bürobereich* kommen, konzentrieren sich meist sehr stark auf den reinen "People Workflow" (siehe Abb. 2). Sie benutzen die graphische Prozessmodellierung vor allem, um den Lauf einer elektronischen Umlaufmappe zu beschreiben und die jeweiligen Bearbeiter (meist ausgedrückt durch "Rollen") von Prozessschritten festzulegen. Die Bearbeiter eines Prozessschrittes erhalten in der Regel eine elektronische Umlaufmappe bereitgestellt, welche die zu bearbeitenden Dokumente enthält. Diesen Dokumenten sind Anwendungen zu deren Bearbeitung zugeordnet (Word, WordPerfect, Excel, Lotus, ...). Die Daten wandern hier, logisch gesehen, zusammen mit den Anwendungen von Benutzer zu Benutzer. Einfache Abweichungen vom vorgeplanten Ablauf (z.B. Einfügen eines neuen Bearbeitungsschrittes vor Weitergabe der Umlaufmappe) sind bei dieser Art von Systemen meist möglich. Hingegen ist die Integration von Anwendungssystemen (wie bei EAI) oder auch die Möglichkeit zur parallelen Ausführung von Prozessschritten nicht oder nur eingeschränkt vorhanden.

Abb. 2: Benutzersicht auf ein prozessorientiertes Informationssystem

Bei Lösungsangeboten aus dem Bereich "*Production Workflow*" stehen vor allem stark standardisierte Abläufe im Fokus [2]. Diesbezüglich besteht eine enge Verwandtschaft zu den Lösungen aus dem EAI-Bereich. Einige dieser Systeme besitzen dementsprechend auch eine Vielzahl von Konnektoren für die Ankopplung von Anwendungssystemen. Um die mit Prozessschritten verknüpften Anwendungen mit den entsprechenden Aufrufdaten zu versorgen, werden üblicherweise die Datenflüsse zwischen den Prozessschritten explizit modelliert. Die Systeme dieser Kategorie zielen aber auch auf "People Workflow" ab und können daher ebenfalls Bearbeiterzuordnungen für Prozessschritte vornehmen. D.h. sie sind in der Lage, "gemischte" Anwendungsszenarien zu bedienen. In dieselbe Richtung zielen Systeme für die prozessorientierte Komposition und Orchestrierung von Web Services (siehe [3]). – Im Gegensatz zu den Lösungen aus dem Verwaltungsbereich (siehe oben), kann bei Systemen dieser Kategorie in der Regel aber nicht oder nur in stark eingeschränktem Maße vom vorgeplanten Ablauf abgewichen werden.

Die beschriebene Situation ist aus Anwendersicht mehr als unbefriedigend. Keines der Systeme deckt das gesamte Spektrum von Anforderungen in zufriedenstellender Weise ab. Insbesondere die für die

Anwendungsintegration geeigneten Systeme resultieren in sehr starren Prozessen und erfordern deshalb einen sehr hohen Prozesserfassungs-, Prozessmodellierungs- und Prozessimplementierungsaufwand, da alle möglichen Ausnahmesituationen antizipiert und geeignet implementiert werden müssen.

Technologisch erinnert diese Situation etwas an die Datenbanksysteme der ersten Generation (IMS, UDS, IDMS, Total, usw.). Die Datenbankschemata werden bei diesen Systemen mittels eines Schemagenerators mit allen Objekttypen, Beziehungen, Zugriffspfaden usw. physisch komplett erzeugt. Ein späteres Hinzufügen eines neuen Schemaobjektes (z.B. eines neuen Recordtyps) erfordert ein Entladen der Datenbank, das Generieren eines neuen Schemas und das Laden der Datenbank in das neue Schema. Auch die Datenbankoperationen in den Anwendungsprogrammen reflektieren nach dem Übersetzen relativ stark die physischen Speicherstrukturen. Eine Änderung an den Speicherstrukturen der Datenbank erfordert deshalb oftmals zumindest auch ein Neuübersetzen der Anwendungsprogramme, in vielen Fällen sind sogar Änderungen an deren Quellcode erforderlich. – Die relationale Datenbanktechnologie veränderte diese Situation in signifikanter Weise. Die deutlich erhöhte Abstraktion von physischen Speicherstrukturen und die Verwendung semantisch hoher, formal fundierter Datenbankoperationen brachten einen großen Gewinn an Flexibilität. Zudem vereinfachten sie den Einsatz und die Wartung von Datenbanken in ganz erheblicher Weise.

Es ist unschwer vorauszusagen, dass prozessorientierten Informationssystemen erst dann der eigentliche Durchbruch gelingen wird, wenn sich eine ähnliche Verbesserung der Situation wie bei den Datenbanksystemen einstellt. Dass ein solcher Fortschritt möglich ist und wie er konkret aussehen könnte, soll im Folgenden anhand von Beispielen aus dem ADEPT- und AristaFlow-Projekt [1, 4] aufgezeigt werden. Wir betrachten die technologischen Konzepte für prozessorientierte Informationssysteme in Kapitel 2 zunächst aus Anwendersicht. Danach gehen wir in Kapitel 3 auf die Technologie als solche ein. Der Beitrag schließt mit einer Zusammenfassung und einem Ausblick in Kapitel 5.

2 Herausforderungen

2.1 Rasche und kostengünstige Realisierung prozess-orientierter Informationssysteme

Die Globalisierung der Märkte und der damit verbundene, zunehmende Konkurrenzdruck zwingen Firmen zu einer immer rascheren Anpassung an sich verändernde Gegebenheiten. Dies zieht sehr oft auch Änderungen in den betrieblichen Abläufen nach sich, wie das Hinzufügen von Aufgaben oder Änderungen bezüglich organisatorischer Strukturen und Abläufe. Die Fähigkeit, rasch Anpassungen vornehmen zu können, ist insbesondere im Kontext von E-Business äußerst wichtig. Hierzu schrieb die Gartner Group bereits im Jahr 1999 [5]: "*Creating e-business processes without a vision for workflow is short-sighted and expensive*" sowie "*E-business time-to-market needs process cloning*".

Es ist also die Fähigkeit gefordert, rasch neue Prozesse realisieren zu können, und zwar entweder durch die Komposition vorhandener Anwendungsfunktionen zu einer neuen Prozessvorlage oder durch Verwendung einer existierenden Vorlage, die man an die neuen Erfordernisse anpasst ("Process Cloning"). Hierbei muss natürlich vermieden werden, dass das Einführen eines neuen Prozesses zu Fehlern bei den bereits vorhandenen ("alten") Prozessen führt. Dieses Risiko besteht sehr real, wenn die Prozesslogik in Form von Zustandsautomaten bzw. Entscheidungstabellen implementiert wird. Dann bedeutet die Einführung eines neuen Prozesses (d.h. eines neuen Prozessschemas bzw. eines neuen Prozesstyps), dass an dem bestehenden Quellcode für den Zustandsautomaten bzw. die Entscheidungstabelle entsprechende Änderungen vorgenommen werden müssen. (Dass damit auch keine neuen Prozesse "on the fly" realisiert werden können, ist ein weiterer Nachteil.)

Abb. 3 zeigt, wie zukünftig die Entwicklung prozessorientierter Informationssysteme aussehen könnte: Zunächst wird eine neue Prozessvorlage erzeugt oder eine bereits vorhandene aus dem Prozessvorlagen-Repository ausgewählt und entsprechend den Erfordernissen angepasst. Danach werden aus dem Komponenten-Repository die Anwendungsfunktionen (z.B. "Auftrag einbuchen", "Material bestellen", "Fertigung planen" etc.) per Drag & Drop an der gewünschten Stelle in die Prozessvorlage eingefügt.

Hierdurch wird die Ausführungsreihenfolge (sequentiell, parallel, entweder-oder) dieser Anwendungsfunktionen festgelegt.

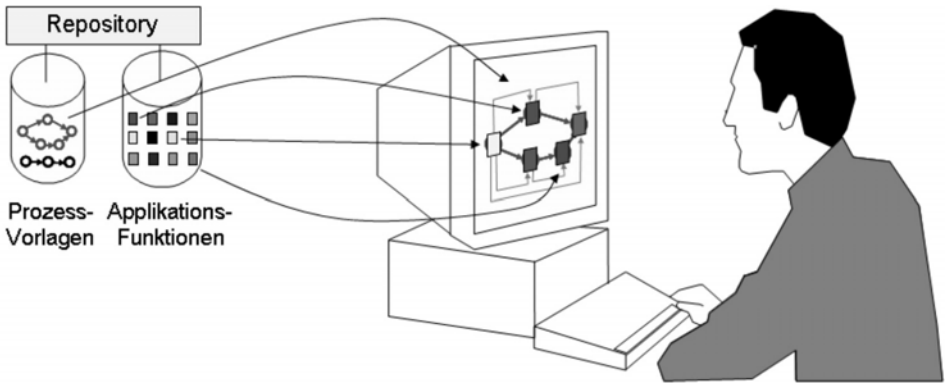


Abb. 3: Realisierung prozessorientierter Informationssysteme mittels "Plug & Play"

Ist dies erledigt, analysiert das System, ob es die Anwendungsfunktionen in dieser Reihenfolge "verschalten" kann. D.h. es ermittelt die zwischen diesen Funktionen existierenden Datenflüsse und sonstige Abhängigkeitsbeziehungen [6] und analysiert, ob diese unter allen möglichen Prozessausführungen eingehalten werden. (Verwandte Problemstellungen treten bei der Komposition von Web Services auf, siehe z.B. [3, 7, 8].) Darüber hinaus sollten weitere Korrektheitsprüfungen erfolgen, etwa in Bezug auf das Vorhandensein unerlaubter Zyklen, Verklemmungen (Deadlocks), Nichterreichbarkeit von Zuständen usw. (Anmerkung: Obwohl sich viele existierende Workflow-Systeme auf Petri-Netze berufen, wo derartige Prüfungen möglich wären, werden diese oft nicht oder zumindest nicht umfassend durchgeführt. "Böse Überraschungen" zur Laufzeit sind daher nicht ausgeschlossen.)

Es ist offensichtlich, dass mit einer solchen Vorgehensweise, die natürlich ein geeignetes Prozess- und Komponenten-Repository voraussetzt, die Realisierung neuer Prozesse um Größenordnungen gegenüber der konventionellen Vorgehensweise (z.B. Implementierung als Zustandsautomat oder Entscheidungstabelle) beschleunigt werden kann.

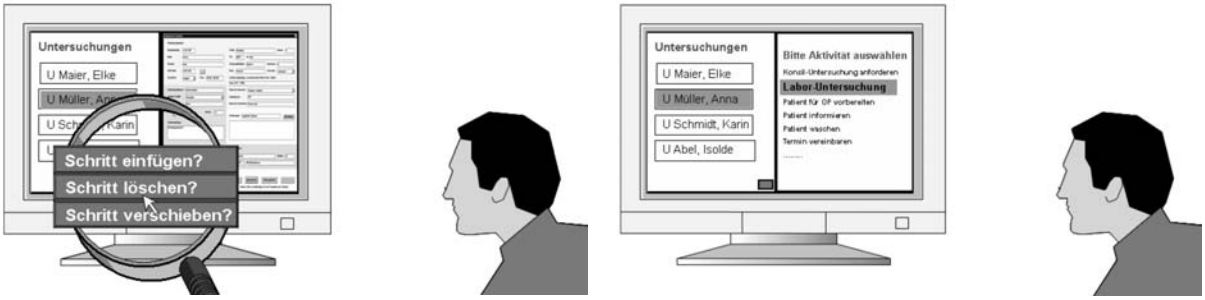
2.2 Flexible Reaktion in Ausnahmesituationen

Offensichtlich ist die Realisierung von prozessorientierten Informationssystemen im Stile von "Plug & Play" bzw. "Drag & Drop" (vgl. Abschnitt 2.1) sehr attraktiv. Wie jedoch bereits erwähnt, erlauben heutige Systeme entweder keine Abweichung vom modellierten Ablauf oder sie erlauben dies zwar, sind dann aber bzgl. der Fähigkeit zur Integration von Anwendungssystemen stark eingeschränkt. (Einige Systeme gestatten Abweichungen, aber es liegt völlig in der Verantwortung des Benutzers oder des Anwendungsentwicklers, ob es dadurch in der Folge zu Systemfehlern kommt oder nicht.) Es wäre natürlich fatal, wenn die Einführung von prozessorientierten Informationssystemen dazu führt, dass die Unternehmen nicht mehr flexibel auf Ausnahmesituationen reagieren könnten.

Zukünftige Prozess-Management-Systeme (als Implementierungsbasis für prozessorientierte Informationssysteme) müssen deshalb in der Lage sein, auch solche Abweichungen vom normalen Ablauf zuzulassen, die nicht bereits in den Prozessvorlagen antizipiert worden sind. Die Durchführung solcher Abweichungen muss so einfach gehalten sein, dass sie vom Endbenutzer (eine entsprechende Autorisierung vorausgesetzt), selbst durchgeführt werden kann. In Abb. 4a bis 4h ist dargestellt, wie eine solche Interaktion aussehen könnte. Die dargestellte Interaktion ist dem ADEPT-System nachempfunden, das die entsprechende Funktionalität aufweist (siehe [9, 10]).

a) Eine Ausnahmesituation tritt auf

b) Benutzer betätigt den "Ausnahmeknopf"

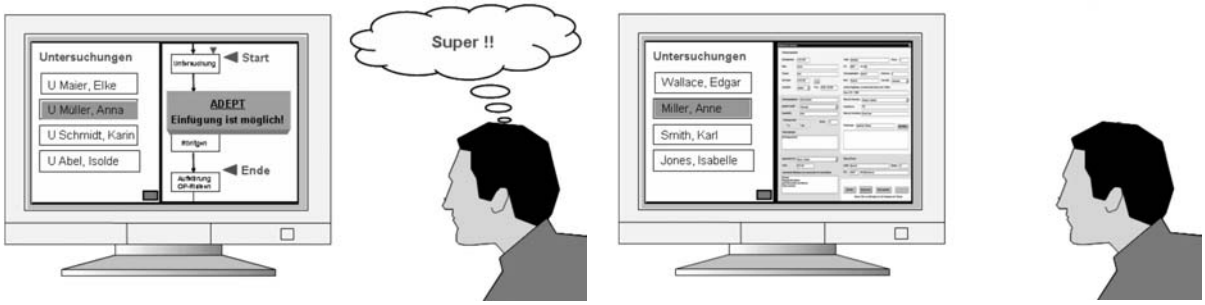


c) Benutzer wählt Art der Änderung

d) Benutzer wählt einzufügenden Schritt

e) Die Einbindung des Schrittes wird festgelegt

f) System prüft, ob Änderung zulässig



g) Die Änderung kann durchgeführt werden

h) Benutzer setzt Tätigkeit fort

Abb. 4: Durchführung einer Ad-hoc-Abweichung aus Benutzersicht

In diesem Beispiel wird unterstellt, dass eine zusätzliche Laboruntersuchung benötigt wird, die in der Prozessvorlage an dieser Stelle im Prozess nicht vorgesehen ist (Abb. 4a). Es wird daher – wenn der Änderungswunsch akzeptiert werden kann – die laufende Prozessinstanz (und nur diese!) individuell verändert. Nachdem der Benutzer den "Ausnahmeknopf" betätigt hat (Abb. 4b), kann er die Art der gewünschten Ad-hoc-Änderung angeben (Abb. 5c). Soll eine Einfügung vorgenommen werden, werden die zur Verfügung stehenden Anwendungsfunktionen angezeigt (Abb. 5d). Dies können einfache Funktionen wie "Brief schreiben" oder "E-Mail versenden" sein, aber auch komplexe Anwendungsdienste. Der Benutzer muss jetzt noch angeben, nach welchem Schritt (bzw. welchen Schritten) die neue Prozessaktivität dem zuständigen Bearbeiter angeboten werden soll und vor welchem Schritt (bzw. vor welchen Schritten) ihre Bearbeitung abgeschlossen sein muss (Abb. 5e). Nach Abschluss der Änderungen prüft das System, ob diese zulässig sind (Abb. 5f und 5g). Im Prinzip werden wieder die gleichen Korrektheitsprüfungen wie zur Modellierungszeit durchgeführt. Auch hier muss wieder

als oberstes Gebot gelten: "Keine bösen Überraschungen zur Laufzeit!". – Wichtig ist, dass diese Funktionalität über das Anwendungs-Programm-Interface (API) angeboten wird und dass sie einfach anzuwenden ist. Eine direkte Manipulation interner Systemzustände durch den Benutzer (mit all den damit verbundenen Fehlermöglichkeiten) muss unbedingt vermieden werden.

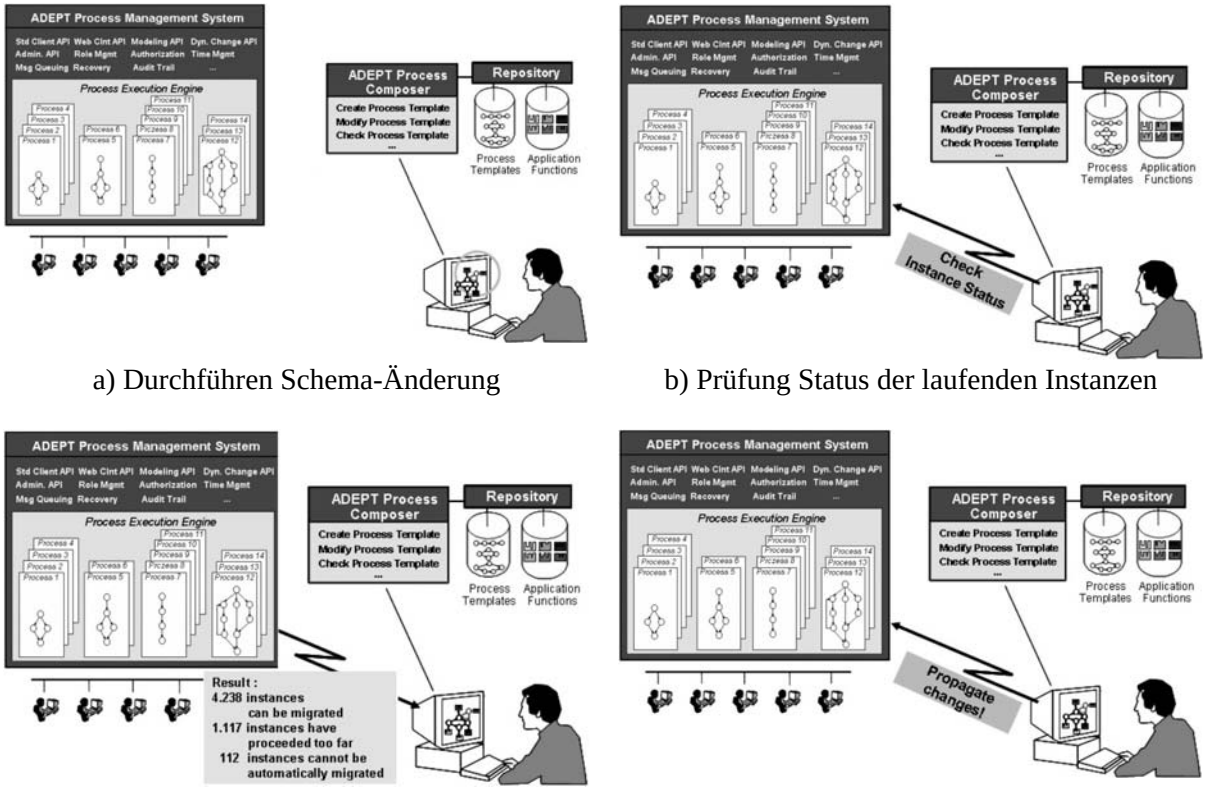
Die Fähigkeit eines Prozess-Management-Systems, Ad-hoc-Abweichungen in der oben beschriebenen Art in kontrollierter und sicherer Weise zu ermöglichen, eröffnet ganz neue Möglichkeiten und setzt auch die Hemmschwelle für den Einsatz von prozessorientierten Informationssystemen in ganz erheblicher Weise herab. Sie ermöglicht zum einen, auf nicht vorhergesehene Sonderfälle *innerhalb* des prozessorientierten Informationssystems zu reagieren (und nicht am System vorbei, wie oftmals heute erforderlich). Zum andern ermöglicht sie, eventuelle Fehler in der Modellierung mittels Ad-hoc-Abweichung zu "reparieren". Darüber hinaus eröffnet sich die Möglichkeit, (seltene) Ausnahmefälle eventuell überhaupt nicht mehr im Prozessmodell zu berücksichtigen, sondern diese – über ein entsprechendes Anwendungsprogramm – nur noch über Ad-hoc-Änderungen zu realisieren. Man könnte sich hier z.B. den Aufbau von Fallbasen vorstellen, die den Benutzern helfen, in Ausnahmesituationen auf bereits bekannte und bewährte Handlungsmuster zurückzugreifen (siehe z.B. [11, 12]).

2.3 Änderungen am (Geschäfts-)Prozess

Die Möglichkeit, Prozesse im Einzelfall (d.h. auf Instanzebene) zu ändern, ist zwar bereits ein großer Schritt in die richtige Richtung, reicht aber alleine noch nicht aus. Muss ein (Geschäfts-) Prozess insgesamt geändert werden, z.B. weil sich die Abteilungsstruktur geändert hat oder zusätzliche Prozessschritte eingefügt werden müssen, dann sind hiervon im Prinzip alle laufenden Instanzen dieses Prozessschemas bzw. dieser Prozessvorlage betroffen; und das können hunderte bzw. tausende von Instanzen sein. Bei kurz laufenden Prozessen wird man in der Regel der Einfachheit halber die bereits laufenden Instanzen noch nach dem alten Schema zu Ende führen. Bei lang laufenden Prozessen oder im Falle gravierender Probleme mit dem bisherigen Ablauf ist dies nicht möglich. Ein jeweils individuelles Anpassen der laufenden Instanzen an das neue Schema mittels Ad-hoc-Änderungen (wie oben beschrieben) wäre im Prinzip zwar möglich, dürfte in der Regel aber zu aufwendig und zu fehlerträchtig sein. Ein zukünftiges Prozess-Management-System sollte daher die Möglichkeit bieten, Änderungen am Prozessschema – falls gewünscht – auf die laufenden Prozessinstanzen zu propagieren, d.h. diese Instanzen auf das neue Prozessschema zu "migrieren".

In Abb. 5a bis 5c ist der Ablauf einer solchen Prozess-Schemaevolution aus Benutzersicht dargestellt. Die Darstellung ist der im Rahmen des ADEPT-Projektes entwickelten Lösung [13, 14] nachempfunden. Eine neue Version des Prozessschemas wird erzeugt und das System berechnet aus den Veränderungsoperationen sowie durch strukturelle Vergleiche (für Details siehe [13]) die durchgeführten Änderungen und leitet aus diesen die Vorbedingungen und die Anpassungsmaßnahmen für die laufenden Prozessinstanzen dieses Typs ab, soweit diese auf das neue Schema migriert werden können (und sollen).

Ad-hoc-Änderungen und Instanz-Migrationen sollten sich hierbei nicht gegenseitig ausschließen. Beide werden für die Unterstützung langlaufender Prozesse benötigt! Beim ADEPT-Ansatz ist dies kombinierte Handhabung dieser beiden Änderungsarten entsprechend realisiert. D.h. es werden, wo dies zu keinen Inkonsistenzen führt, auch individuell veränderte Prozessinstanzen auf das neue Prozessschema migriert und dieses ggf. auf Instanzebene gleich wieder entsprechend individuell modifiziert. Alle mit der Migration zusammenhängenden Status- und Konsistenzprüfungen werden (natürlich) wieder systemseitig vorgenommen. Weitere Details sowie eine Diskussion verschiedener Ansätze finden sich in [13, 15].



a) Durchführen Schema-Änderung

b) Prüfung Status der laufenden Instanzen

c) Ergebnis der Prüfung

d) Durchführung Instanz-Migration

Abb. 5: Prozess-Schemaevolution mit Propagation der Änderungen auf laufende Instanzen

Insbesondere die Kombination aller drei Aspekte, Prozesskomposition mittels Plug & Play, Ad-hoc-Abweichungen zur Laufzeit sowie Prozess-Schemaevolution eröffnen für die kontinuierliche Prozessverbesserung (siehe Abb. 6) ganz neue Perspektiven. Plug & Play ermöglicht die rasche Realisierung neuer sowie die programmtechnische Änderung existierender Prozesse. Während der Prozessausführung fallen Protokolldaten an, die mit geeigneten Werkzeugen analysiert und ausgewertet werden können (siehe z.B. [16]). Bei heutigen Workflow- bzw. Prozess-Management-Systemen kann man eigentlich nur feststellen, ob die der Modellierung zugrunde liegenden Annahmen über Ausführungshäufigkeiten sowie Dauern von Prozessinstanzen und Prozessschritten der Realität entsprechen. Ob der Prozess als solcher evtl. nicht optimal gestaltet ist, lässt sich aus den Protokolldaten nur indirekt oder sehr eingeschränkt (z.B. nie durchlaufene Pfade) herauslesen. Unterstützt man hingegen Ad-hoc-Abweichungen zur Laufzeit, so schafft man damit eine Möglichkeit, ungünstig oder fehlerhaft gestaltete Prozesse vor Ort zu korrigieren (wie man das bei einer manuellen Prozessausführung sehr häufig auf informelle Weise auch machen würde). Durch Beobachtung solcher Abweichungen kann man sehr viel aussagefähigere Informationen über Probleme und Mängel des jeweiligen Prozessmodells erhalten. Mittels Prozess-Schemaevolution können diese Änderungen dann auf einfache und kostengünstige Weise auf die laufenden Instanzen übertragen und somit sofort effektiv werden [17].

2.4 Weitere Herausforderungen

Prozessorientierte Informationssysteme der nächsten Generation sollten noch eine Reihe weiterer Fähigkeiten besitzen, etwa die Überwachung zeitlicher Vorgaben (Deadlines sowie Mindest- und Maximalzeitabstände zwischen Prozessschritten), die Mitbetrachtung des Ressourcenbedarfs, das intelligentes Scheduling von Prozessschritten (unter Beachtung dieser Aspekte), die systemseitige Unterstützung beim kompletten oder partiellen Zurücksetzen von Prozessen im Fehlerfall und vieles mehr. Hierauf können wir aus Platzgründen in diesem Beitrag nicht tiefer eingehen.

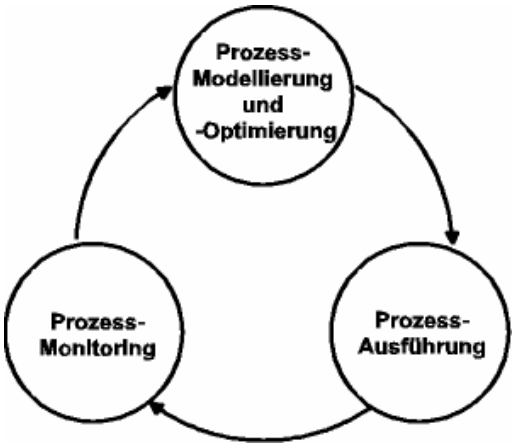


Abb. 6: Zyklus der kontinuierlichen Prozessverbesserung

3 Herausforderungen aus technologischer Sicht

Die größte technologische Herausforderung besteht darin, die in den Abschnitten 2.1 bis 2.4 beschriebenen Herausforderungen in ihren Wechselwirkungen zu verstehen und auf dieser Basis dann geeignete Lösungen zu finden, die auch im Zusammenspiel funktionieren. Dies erfordert zwingend ein adäquates formales Rahmenwerk, um Vor- und Nachbedingungen für Prozessänderungen definieren zu können sowie entsprechende Umformungsregeln formalen Korrektheitsnachweisen zugänglich zu machen.

Eine zentrale Rolle spielen in diesem Zusammenhang das *Prozess-Metamodell*, die formale Repräsentation der daraus abgeleiteten Prozessinstanzen sowie die interne Repräsentation von Prozessschemata und -instanzen im Prozess-Management-System zur Ausführungszeit. Das Prozess-Metamodell legt zum einen fest, mit welchen syntaktischen Konstrukten Prozesse formal beschrieben und ob alle relevanten Ablaufmuster ("Workflow Patterns" [18]) ausgedrückt werden können oder ob hier Einschränkungen bestehen. Zum andern bestimmt das Prozess-Metamodel (und seine formale Fundierung) in ganz entscheidender Weise, in welchem Umfang Korrektheitsprüfungen und –aussagen möglich sind. So scheint es auf den ersten Blick egal zu sein, ob man für die Modellierung von UND- und XOR-Verzweigungen verschiedene Knotentypen verwendet oder ob man hierfür denselben Knotentyp benutzt und durch Transitionsprädikate an den Kanten steuert, ob sich mehrere Pfade oder nur ein Pfad für die Ausführung qualifizieren. Sind diese Prädikate nicht statisch entscheidbar, kann erst zur Laufzeit festgestellt werden, welcher Verzweigungstyp vorliegt. Dies wiederum schränkt ganz erheblich die Möglichkeiten ein, das Prozessmodell bereits zur Modellierzeit auf Konsistenz (z.B. alle Datenflüssen korrekt, keine Deadlocks) zu prüfen. Ähnliches gilt, wenn das Prozess-Metamodell kein Schleifenkonstrukt kennt und somit z.B. nicht zwischen gewünschten Schleifen und unerwünschten Zyklen unterschieden werden kann.

Ein anderer nicht ganz unwesentlicher Aspekt des Prozess-Metamodells ist, ob die Korrektheitsprüfungen effizient durchgeführt werden können. Dieser Aspekt spielt zwar bei der Modellierung noch keine entscheidende Rolle, wird aber sehr wichtig, wenn Ad-hoc-Abweichungen zur Laufzeit möglich sein sollen. Sollen für diese ebenfalls wieder die Konsistenzbedingungen wie zur Modellierungszeit gelten (was sehr anzuraten ist, ansonsten drohen unangenehme "Überraschungen" zur Laufzeit), dann ist es für die Antwortzeit von entscheidender Bedeutung, ob der zu analysierende Bereich auf dem Prozessgraphen der betroffenen Instanz eingeschränkt werden kann oder ob hierfür eine komplette Graphanalyse durchgeführt werden muss.

Die Aspekte *Ausdrucksmächtigkeit*, *Korrektheitsüberprüfungen* und *effiziente Durchführung* (insbesondere nach Änderungen) stehen teilweise in Zielkonflikt zueinander. Die Wahl für ein bestimmtes Prozess-Metamodell sowie die Repräsentation einer Prozessinstanz stellen deshalb immer einen Kom-

promiss dar. "Das" beste Metamodell und die beste interne Repräsentation für Instanzen gibt es nicht. Je nach Gewichtung der Aspekte wird man hier zu jeweils einer anderen Lösung kommen.

Das ADEPT-Metamodell [9, 10] (siehe Abb. 7) stellt ebenfalls einen solchen Kompromiss dar. Hinsichtlich der Anforderungen standen die möglichst adäquate Beschreibung praxisrelevanter Prozessmuster, deren formale Überprüfbarkeit, die Eignung (im Wesentlichen) auch als formales Modell für die Prozessinstanzen und, in diesem Zusammenhang, die Eingrenzbarkeit des Analysebereichs auf dem Prozessgraphen für ein möglichst großes Spektrum von Ad-hoc-Änderungen an oberster Stelle auf der Prioritätenliste. In Kauf genommen wurde, dass einige Prozessmuster etwas umständlicher modelliert werden müssen, als dies bei einem "liberaleren" Prozess-Metamodell (wie z.B. Petrinetzen) der Fall ist. Herausgekommen ist ein blockorientiertes Prozess-Metamodell mit syntaktisch unterschiedlichen Verzweigungsknoten, mit Schleifenkonstrukten, mit Rücksprungkanten zur Modellierung bekannter Ausnahmefälle, mit zusätzlichen Synchronisationskanten, um die Beschränkungen der Blockstruktur bei Bedarf zu kompensieren, sowie mit Prozessvariablen und Datenflusskanten.

Prozessmodelle, die auf dem ADEPT-Metamodell basieren, sind – wenn der Modelleditor syntaxgesteuert arbeitet – schon weitgehend "per Konstruktion" korrekt, so dass sich die Analyse auf einige wenige Aspekte (z.B. Datenflüsse und Synchronisationskanten) beschränken kann. Ebenso kann bei solchen Prozessmodellen im Kontext von Ad-hoc-Änderungen der zu analysierende Bereich oftmals stark eingeschränkt werden. Die Änderung selbst läuft sehr einfach und schematisch ab. Der Ablauf einer Einfügung und die damit verbundenen Änderungen am (logischen) Prozess-Instanzgraphen sind in Abb. 8a bis 8d anhand eines Beispiels illustriert. Abb. 8a zeigt die Situation, nachdem der Benutzer (oder das Anwendungsprogramm) bestimmt hat, welcher Prozessschritt eingefügt werden soll (hier: "perform allergy test") und nach welchem bzw. vor welchen Prozessschritten der neue Prozessschritt abgeschlossen sein soll. Dies entspricht dem in Abb. 4c bis 4e beschriebenen Szenario, nun von der Systemseite her betrachtet. Nachdem die Rahmenbedingungen spezifiziert wurden, wird der sog. "minimale Block" bestimmt, der den gesamten Einfügebereich umfasst und der neue Prozessschritt wird (zunächst) parallel zu diesem Block eingefügt (siehe Abb. 8b). Anschließend werden zwischen dem "Vorbereich" und dem "Nachbereich" des neuen Schrittes Synchronisationskanten eingefügt, um die gewünschte Ausführungsreihenfolge zu erreichen (siehe Abb. 8c). Durch entsprechende Vorbedingungen bzw. Analysen ist sichergestellt, dass hierdurch keine (unerlaubten) Zyklen entstehen. Außerdem wird geprüft, ob die Datenversorgung des neuen Schrittes durch den Prozesskontext gewährleistet ist oder ob hierfür ggf. Hilfsschritte einzufügen sind.

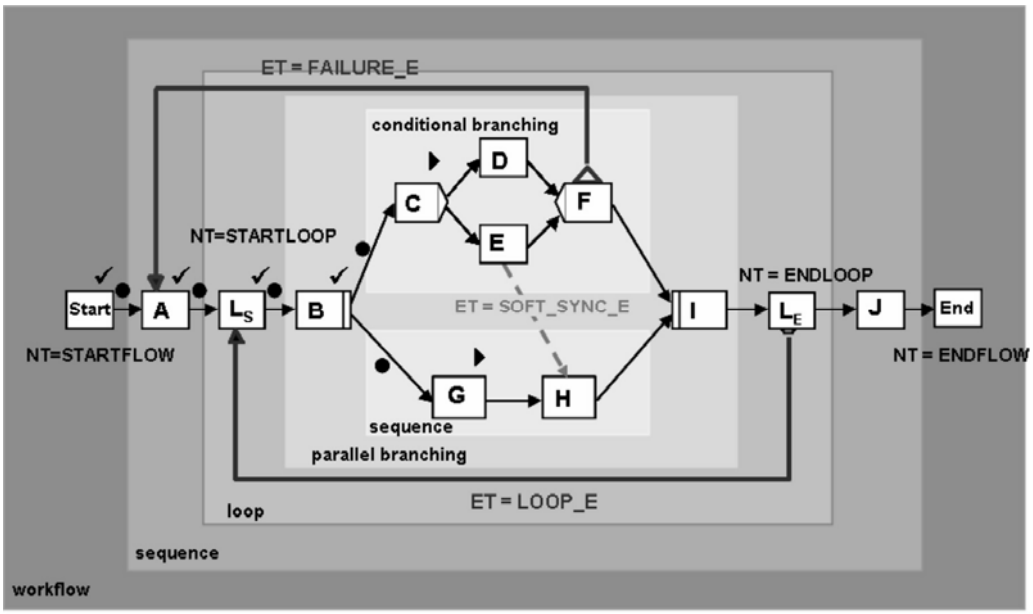


Abb. 7: ADEPT Prozess-Metamodell

Zeigen diese Prüfungen, dass die Einfügung dieses Schrittes zu keinen Konsistenzproblemen führt, wird versucht, ob einige der Kanten und Knoten durch die Anwendung von Vereinfachungsregeln eliminiert bzw. durch kompaktere Konstrukte ersetzt werden können. Anschließend wird der neue Schritt mit dem Datenkontext verbunden (siehe Abb. 8d) und dem Benutzer bzw. dem Anwendungsprogramm der Vollzug der Prozessinstanzänderung gemeldet.

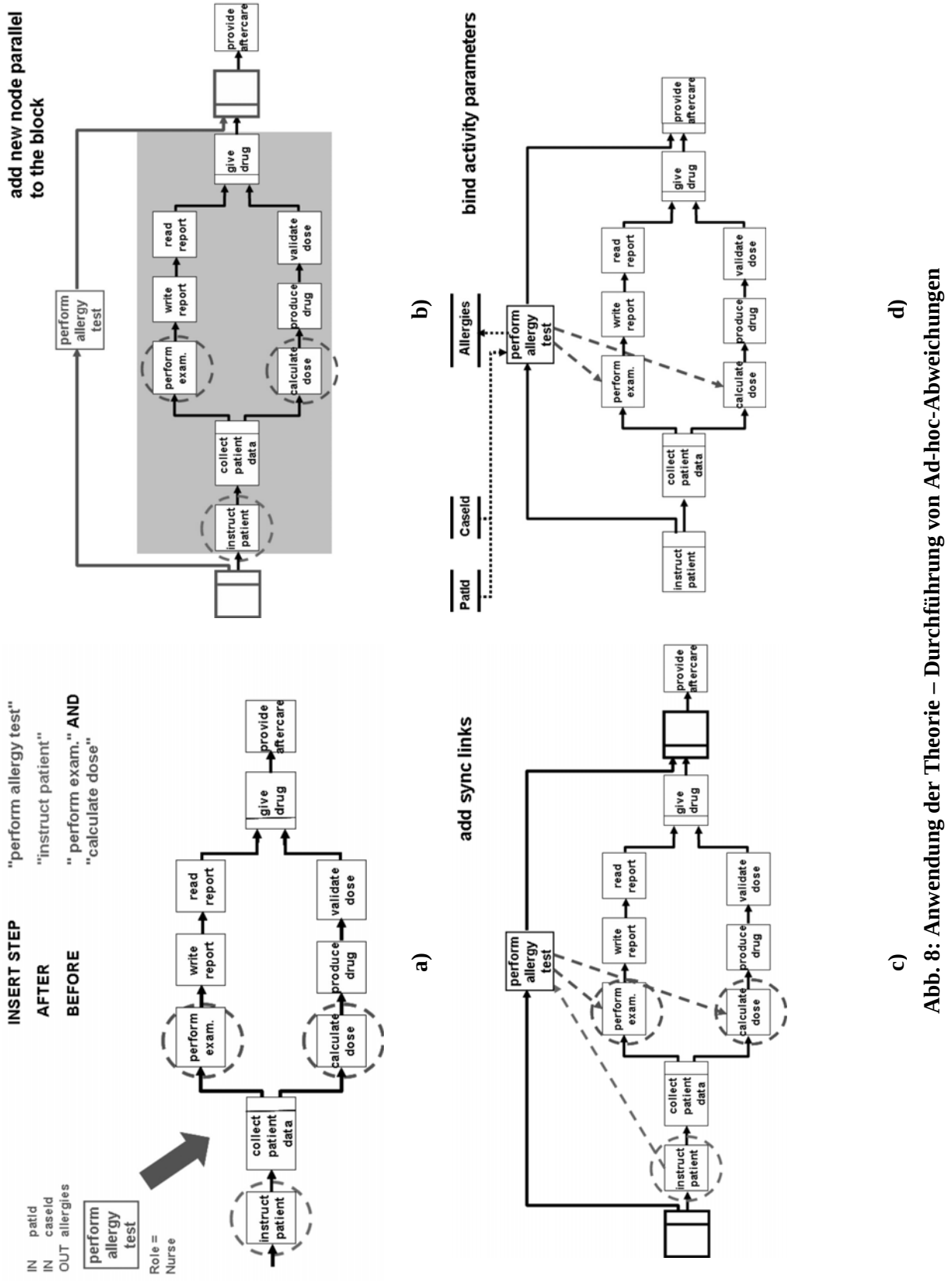


Abb. 8: Anwendung der Theorie – Durchführung von Ad-hoc-Abweichungen

Das ADEPT-Metamodell hat sich auch bei der Entwicklung der Verfahren zur *Prozess-Schemaevolution* als sehr nützlich erwiesen. Dadurch, dass die Ablaufhistorie durch Markierungen im Graph festgehalten wird (siehe Abb. 8; diese Information wird auch für Korrektheitsprüfungen bei Ad-hoc-Änderungen benötigt), entfällt in der Mehrzahl der Fälle ein aufwendiges Analysieren der Prozesshistorie durch Zugriff auf die Protokolldaten der Prozessinstanzen.

Im Prinzip könnte man die Prozess-Schemaevolution als einen Spezialfall einer Ad-hoc-Änderung betrachten und diese dadurch realisieren, dass man die Änderungen am Prozessschema wie eine Ad-hoc-Änderung interpretiert und versucht, diese jeweils auf die Prozessinstanz anzuwenden. Gelingt, dies gilt die Prozessinstanz als auf das neue Schema migriert. Gelingt dies wegen Konsistenzproblemen nicht, wird die Instanz als nicht migrierbar eingestuft und verbleibt auf dem alten Schema. Diese Vorgehensweise verbietet sich jedoch zum einen aus Aufwandsgründen und würde auch alle Prozessinstanzen, die durch Ad-hoc-Änderungen individuell modifiziert wurden, von einer Migration ausschließen.

Aus diesem Grund wurden im Rahmen des ADEPT-Projektes spezielle Verfahren entwickelt, um die Prozessinstanzen hinsichtlich Status sowie ggf. der Beziehung zwischen Ad-hoc- und Schemaänderung (z.B. überlappend oder nicht überlappend [21]) zu klassifizieren und darauf zugeschnittene Migrationsmethoden anzuwenden. Diese Verfahren sind weitgehend auch auf andere Prozess-Metamodelle übertragbar, sind jedoch bei ADEPT-ähnlichen Metamodellen besonders leistungsfähig (siehe [13, 15] für nähere Details).

Die Realisierung von prozessorientierten Informationssystemen mittels Plug & Play stellt sowohl für die Entwurfs- und Entwicklungsumgebung als auch für die Ausführungsumgebung (das Prozess-Management-System) eine große Herausforderung dar. Im Bereich der Entwurfs- und Entwicklungsumgebung für Anwendungskomponenten ergeben sich unter anderem folgende Fragestellungen:

- Welche Arten von Abhängigkeiten zwischen Anwendungsfunktionen und Zuständen von Komponenten müssen modelliert und im Komponenten-Repository hinterlegt werden?
- Wo kommen diese Information her? Wie beschreibt man sie?
- Wie müssen die Komponenten realisiert sein, damit sie in verschiedenem Anwendungskontext einsetzbar sind?

Bei der Prozessmodellierung (und auch auf der Prozessinstanzebene) müssen diese Informationen ins Prozessmodell integriert werden, um diese Aspekte bei der Evaluierung des Modells mit berücksichtigen zu können, und zwar auch im Kontext von Ad-hoc-Abweichungen.

4 Zusammenfassung und Ausblick

Prozess-Management-Systeme müssen gegenüber dem heutigen Stand erheblich leistungsfähiger werden, damit prozessorientierte Informationssysteme für ein breites Spektrum von Anwendungen einsetzbar werden. Ein besonders kritischer Punkt in diesem Zusammenhang ist die Realisierung einer höheren Flexibilität und Adaptivität der Systeme. Wir haben zunächst aus Anwendersicht aufgezeigt, wie der Umgang mit Systemen dieser Art aussehen könnte und sind danach auf die daraus resultierenden technologischen Herausforderungen eingegangen. Wir haben hierbei teilweise auf die im ADEPT-Projekt erarbeiteten Ergebnisse Bezug genommen, das hinsichtlich dieser Aspekte derzeit die Spitze des technischen Fortschritts markiert.

Die oben beschriebene Ad-hoc-Flexibilität basiert auf den in [9, 10] beschriebenen Konzepten und wurde im ADEPT-System [19, 20] realisiert. Dieses existiert bereits seit dem Jahr 2000 und ist unseres Wissens derzeit (immer noch) das mächtigste System seiner Art. Es wird in einer Reihe von Forschungsprojekten im In- und Ausland eingesetzt. Die beschriebene Funktionalität zur Prozess-Schemaevolution wurde in den letzten Jahren forschungsmäßig im Rahmen des ADEPT-Projektes

sehr intensiv bearbeitet [13, 15, 14] und wurde in einem separaten Prototypen für Evaluierungs- und Demonstrationszwecke ebenfalls implementiert. Beide Funktionalitäten sowie einige weitere Features werden in die Implementierung des ADEPT2-Systems einfließen, d.h. sie werden dann in einem System in integrierter Weise zur Verfügung stehen. Damit ist die Umsetzung des in Abb. 6 skizzierten Zyklus der kontinuierlichen Prozessverbessserung in greifbare Nähe gerückt.

Die oben beschriebene Anwendungskomposition mittels Plug & Play wird derzeit im Rahmen des AristaFlow-Projektes konzipiert und implementiert. Als Laufzeitumgebung wird ADEPT2 zum Einsatz kommen. Nähere Informationen zu den Projekten ADEPT und AristaFlow finden sich unter [1, 4].

5 Literatur

[1] Siehe http:\\www.AristaFlow.de

[2] Leymann, F.; Roller, D.: Production Workflow - Concepts and Techniques. Prentice Hall PTR, 2000

[3] Reichert, M.; Stoll, D.: Komposition, Choreographie und Orchestrierung von Web Services - Ein Überblick. EMISA FORUM, Mitteilungen d. GI-FG "Entwicklungsmethoden für Informationssysteme u. deren Anwendung", Band 24, Heft 2, 2004, S. 21-32

[4] http:\\www.informatik.uni-ulm.de\dbis → Forschung → ADEPT

[5] McCoy, D.: Why E-Business Craves Workflow Technology. Gartner Group, Research Note, Technology, T-09-4929, 16. Dezember 1999

[6] Acker, H.; Atkinson, C.; Dadam, P.; Rinderle, S.; Reichert, M.: Aspekte der komponentenorientierten Entwicklung adaptiver prozessorientierter Unternehmenssoftware. In: Turowski, K. (Hrsg.): Architekturen, Komponenten, Anwendungen - Proc. 1. Verbundtagung AKA 2004, Augsburg, Dezember 2004. LNI P-57, 2004, S. 7-24

[7] Berardi, D.; Calvanese, D.; De Giacomo, G. et al.: Automatic Compostition of E-services That Export Their Behavior. Proc. Int'l Conf. on Service-Oriented Computing-ISOC 2003, Trento, Italy, Dez. 2003, LNCS 2910, S. 43-58

[8] Kaleem, M.F.: A Classification Framework for Approaches and Methodologies to make Web Services Compositions Reliable. Proc. First Europ. Workshop on Object Orientation and Web Service. Darmstadt, IBM Research Report Series-RA 220, 2003, S. 35-42

[9] Reichert, M.; Dadam, P.: ADEPTflex - Supporting Dynamic Changes of Workflows Without Losing Control. Journal of Intelligent Information Systems, Kluwer Academic Publ., Vol. 10, No. 2, March/April 1998, S. 93-129

[10] Reichert, M.: Dynamische Ablaufänderungen in Workflow-Management-Systemen. Dissertation, Universität Ulm, Fakultät für Informatik, Mai 2000

[11] Rinderle, S.; Weber, B.; Reichert, M.; Wild, W.: Integrating Process Learning and Process Evolution - A Semantics Based Approach. Proc. 3rd Int'l Conf. Business Process Management (BPM'05), Nancy, France, September 2005 (accepted for publication)

[12] Weber, B.; Werner, W.; Breu, R.: CCBR-Enabled Adaptive Workflow Management. Proc. European Conf. on Case-Based Reasoning (ECCBR'04), Madrid, 2004

[13] Rinderle, S.: Schema Evolution in Process Management Systems. Dissertation, Universität Ulm, Fakultät für Informatik, Dezember 2004

[14] Rinderle, S.; Reichert, M.; Dadam, P.: On Dealing With Structural Conflicts Between Process Type and Instance Changes. Business Process Management - Proc. 2nd Int'l Conf. BPM 2004, Potsdam, Germany, Juni 2004, LNCS 3080, S. 274-289

[15] Rinderle, S.; Reichert, M.; Dadam, P.: Correctness Criteria for Dynamic Changes in Work-

flow Systems - A Survey. Data and Knowledge Engineering, Vol. 50, No. 1, 2004, Special Issue on Advances in Business Process Management, S. 9-34

[16] van der Aalst, W.; Weijters, T.; Maruster, L.: Workflow Mining: Discovering Process Models from Event Logs. IEEE Transactions on Knowledge and Data Engineering, Vol. 16, No. 9, September 2004, S. 1128-1142

[17] Weber, B.; Reichert, M.; Rinderle, S.; Wild, W.: Towards a Framework for the Agile Mining of Business Processes. Proc. 1st Int'l Workshop on Business Process Intelligence (BPI 2005), held in conjunction with BPM'05, Nancy, September 2005 (to appear)

[18] van der Aalst, W.M.P.; ter Hofstede, A.H.M.; Kiepuszewski, B.; Barros, A.P.: Workflow Patterns. Distributed and Parallel Databases - An International Journal, Vol. 14, Issue 1, Juli 2003, S. 5 – 52

[19] Hensinger, C.; Reichert, M.; Bauer, T.; Strzeletz, T.; Dadam, P.: ADEPTworkflow - Advanced Workflow Technology for the Efficient Support of Adaptive, Enterprise-wide Processes. Proc.. Software Demonstration Track, EDBT'2000 Conf., Konstanz, März 2000, S. 29-30

[20] Reichert, M.; Rinderle, S.; Kreher, U.; Dadam, P.: Adaptive Process Management with ADEPT2. Proc. Int'l Conf. on Data Engineering, ICDE '05, Tokyo, April 2005, Demo Session, S. 1113-1114

[21] Rinderle, S.; Reichert, M.; Dadam, P.: Disjoint and Overlapping Process Changes - Challenges, Solutions, Applications. Proc. 12th Int'l Conf. Cooperative Information Systems (CoopIS'04), Agia Napa, Cyprus, LNCS 3290, Okt. 2004, S. 101-121