M. Glinz, G. Müller-Luschnat (Hrsg.): Proc. Modellierung 2002, Arbeitstagung der GI, Tutzing, März 2002, S. 183-194 (GI-Edition Lecture Notes in Informatics, Band P-12)

Modellierung planbarer Abweichungen in Workflow-Management-Systemen

Manfred Reichert, Thomas Bauer, Thomas Fries, Peter Dadam

Abteilung Datenbanken und Informationssysteme
Universität Ulm, D-89069 Ulm
{reichert, bauer, fries, dadam}@informatik.uni-ulm.de

Abstract: Workflow-Management-Systeme (WfMS) sind eine vielversprechende Technologie für die Realisierung prozessorientierter Anwendungen. Allerdings bieten heutige WfMS keine ausreichende Unterstützung zur Behandlung von Ausnahmen. Im ADEPT-Projekt haben wir deshalb fortschrittliche Modellierungs- und Ausführungskonzepte entwickelt, die auf eine Erhöhung der Flexibilität von WfMS zielen. Sie ermöglichen es zum einen, planbare Abweichungen vom Standardablauf eines Arbeitsprozesses bereits zur Modellierzeit festzulegen, zum anderen können nicht vorhersehbare Abweichungen auch dynamisch zur Laufzeit erfolgen. Dieser Beitrag konzentriert sich auf den erstgenannten Aspekt. Er zeigt auf, wie sich planbare Abweichungen sinnvoll modellieren lassen, welche Anforderungen dabei bestehen und welche Möglichkeiten bzw. Grenzen mit einem solchen Ansatz verbunden sind. Unsere Erfahrung mit konkreten Anwendungen aus dem Krankenhausbereich hat gezeigt, dass entsprechende Modellierungsmöglichkeiten einen wichtigen Beitrag zur Erhöhung der Flexibilität von WfMS leisten.

1 Einleitung

WfMS bieten eine vielversprechende Technologie zur Realisierung prozessorientierter Anwendungssysteme [Ob96]. Charakteristisch für viele WfMS ist die Trennung von Prozesslogik und Anwendungscode [LR00]. Die Ablauflogik der Arbeitsprozesse (engl. *Workflows*; kurz: WF) wird dem WfMS explizit durch die Modellierung bekannt gemacht und nicht im Programmcode „versteckt“. Dieser Ansatz bietet mehrere Vorteile: Die systemseitige Durchführung der Ablaufsteuerung vereinfacht die Anwendungsentwicklung erheblich. Durch die explizite Beschreibung der Prozesslogik kann zudem das zukünftige Systemverhalten vorab evaluiert werden, wodurch sich Entwurfsfehler noch vor Implementierung der eigentlichen Anwendungskomponenten entdecken lassen. Aus denselben Gründen sind spätere Änderungen in den Geschäftsprozessen und daraus resultierende Anpassungen der Anwendungssysteme einfacher durchführbar.

Trotz dieser Vorteile sind WfMS in der betrieblichen Praxis nicht verbreitet. Ein wesentlicher Grund hierfür ist ihre mangelhafte Flexibilität [MR99, We98]. So gestatten es heutige WfMS nur eingeschränkt, vom einmal modellierten Standardablauf abzuweichen (z.B. durch Auslassen von WF-Schritten). Dies ist aber oftmals unerläßlich, um flexibel auf Ausnahmensituationen reagieren zu können. Beispiele für Ausnahmen sind die Verschlechterung des Zustands eines Patienten im Verlauf eines Behandlungsprozesses oder Fehler bei der Herstellung eines Produktes. Ausnahmen beschreiben also Ereignisse, deren Auftreten eine Abweichung vom „normalen“ Ablauf erfordert [SM95].

Generell muss zwischen planbaren und nicht planbaren Ausnahmen bzw. Abweichungen unterschieden werden. Für planbare Ausnahmen ist der Kontext ihres Auftretens ebenso bekannt, wie die zu ihrer Behandlung notwendigen Aktionen. Dementsprechend können diese Ausnahmen bereits bei der WF-Modellierung berücksichtigt werden. Bei nicht planbaren Ausnahmen handelt es sich dagegen um a priori unbekannte Ereignisse, die erst zur Laufzeit behandelt werden können. Zur besseren Differenzierung verwenden wir im Folgenden die Begriffe A-priori- und A-posteriori-Flexibilität [Jo99].

Zur Erhöhung der A-posteriori-Flexibilität bietet ADEPT fortschrittliche Konzepte für dynamische WF-Änderungen, die ausführlich in [RD98, Re00] beschrieben werden. Sie gestatten es zur Laufzeit, die Struktur, den Zustand und die Attribute von WF-Instanzen anzupassen. Zur Sicherung der Konsistenz werden bei der Durchführung solcher Ad-hoc-Änderungen umfangreiche Korrektheitsüberprüfungen vorgenommen [Re00]. Dieser Beitrag behandelt Konzepte zur Verbesserung der A-priori-Flexibilität in WfMS. Motivation hierfür bilden langjährige Erfahrungen mit Krankenhausabläufen [DRK00]. Hier hat sich gezeigt, dass Ausnahmen oftmals a priori bekannt sind und deshalb bereits bei der WF-Modellierung berücksichtigt werden können. Dadurch verringert sich auch die Häufigkeit der vergleichsweise teuren Ad-hoc-Modifikationen zur Laufzeit.

Um die Anforderungen an die Modellierung planbarer Abweichungen besser zu verstehen, sind sowohl die Sicht des Modellierers als auch der späteren Anwender zu beachten. Aus Modellierersicht muss der Regelfall von ausnahmebedingten Abweichungen getrennt beschreibbar sein. Darüber hinaus darf unter der Einbeziehung von Ausnahmebehandlungen die Übersichtlichkeit der WF-Modelle nicht leiden oder sich der Aufwand für ihre Erstellung wesentlich erhöhen. Aus Anwendersicht ist es wichtig, dass „normale“ Aktivitäten von ausnahmebedingten Aktionen, deren Ausführung eine Abweichung vom Standardablauf bedeutet, unterscheidbar sind.

Abschnitt 2 fasst wichtige Grundlagen zusammen, die für das weitere Verständnis nötig sind. In Abschnitt 3 zeigen wir, wie sich planbare Vorwärtssprünge modellseitig abbilden lassen. Abschnitt 4 behandelt planbare Rückwärtssprünge. Es folgen in Abschnitt 5 eine Diskussion verwandter Ansätze und in Abschnitt 6 eine kurze Zusammenfassung.

2 Grundlagen

Für jeden zu unterstützenden Prozesstyp muss eine WF-Modell erstellt und im WfMS hinterlegt werden. Ein solches WF-Modell beschreibt, welche Arbeitsschritte (sog. Aktivitäten) von welchen Benutzern in welcher Reihenfolge und unter Nutzung welcher Daten und Programme bearbeitet werden sollen. Zur Modellierung der verschiedenen WF-Aspekte dient das ADEPT-Basismodell. Es ermöglicht die Überprüfung statischer und dynamischer Modelleigenschaften, etwa im Hinblick auf die Terminierung des WF, die Erfüllbarkeit von Zeitbeschränkungen oder die Vollständigkeit von Datenflüssen.

Für die Modellierung verfolgen wir einen blockbasierten Ansatz, bei dem Sequenzen, Verzweigungen (AND/AND; XOR/XOR; AND /XOR) und Schleifen als Blöcke mit jeweils genau einem Ein- und Ausgangsknoten modelliert werden. Solche Kontrollblöcke können geschachtelt sein, dürfen sich aber nicht überlappen. Um die Ausdrucksmächtigkeit des Metamodells zu erhöhen, umfaßt es zusätzliche Konstrukte, etwa zur

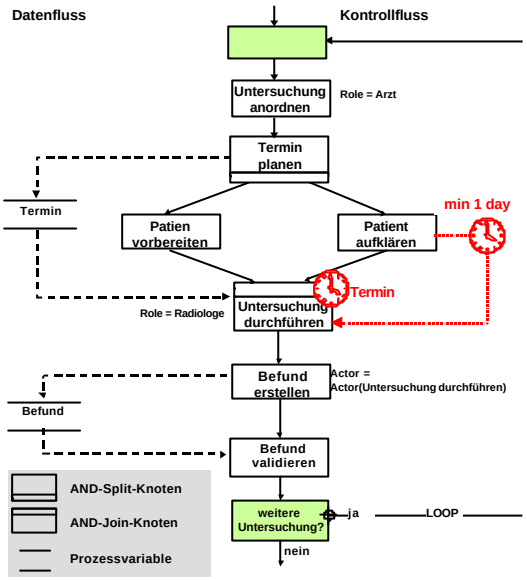

Abb. 1: Workflow-Modellierung in ADEPT

Beschreibung von Wartet-Auf-Beziehungen zwischen Aktivitäten paralleler Zweige. Ein einfaches Beispiel zeigt Abb. 1.

Nach Erzeugung und Start einer WF-Instanz wird diese über ihre komplette Lebensdauer hinweg vom WfMS kontrolliert. Dazu wird ein Ausführungsgraph verwaltet, dessen Knoten und Kanten bestimmte Zustandsmarkierungen besitzen. Es gibt wohldefinierte Regeln, die festlegen, unter welchen Markierungen eine Aktivität aktiviert werden darf und welche Folgemarkierungen sich im Anschluss an ihre Ausführung ergeben. Hierauf basierend steuert das WfMS die WF-Ausführung, bietet die zur Bearbeitung anstehenden Aktivitäten zuständigen Akteuren in Arbeitslisten an, führt automatische Schritte selbständig durch und ruft die für die Ausführung von Aktivitäten passenden Programme (mit den richtigen Daten) auf.

3 Planbare Vorwärtssprünge

Ausnahmebedingt kann es erforderlich werden, die Bearbeitung von Aktivitäten zu überspringen oder Aktivitäten vorzeitig auszuführen (vgl. Bsp. 1).

Beispiel 1: (Überspringen von Aktivitäten):
Die Abwicklung einer medizinischen Untersuchung umfasst mehrere Arbeitsschritte: Anordnung der Untersuchung, Planung des Untersuchungstermins, Vorbereitung und Aufklärung des Patienten, Untersuchungsdurchführung sowie Erstellung und Validation eines Befundes (vgl. Abb. 1). Bereits bei dieser einfachen Prozesskette müssen Benutzer bei Bedarf in flexibler Form vom Standardablauf abweichen können. So muss die Untersuchung im Ausnahmefall auch ohne die ansonsten übliche Terminvereinbarung und ohne die standardmäßigen Vorbereitungen notfallmäßig durchgeführt werden können. D.h. Anwender müssen diese Schritte überspringen können.

Häufig sind solche Abweichungen vom „normalen“ Ablauf planbar. Ist etwa a priori bekannt, dass die Bearbeitung einer Aktivität in Ausnahmefällen vorgezogen werden können muss, so sollte dies im WF-Modell abbildbar sein. Dadurch wird das WfMS in die Lage versetzt, die betreffende Aktivität zur Laufzeit als „Ausnahmeschritt“ anzubieten, ohne dass zu ihrer Auswahl aufwendige Interaktionen (wie im Fall von Ad-hoc-Änderungen) notwendig werden. Wichtig ist, dass die Zielaktivität eines Vorwärtssprungs dem Anwender in einer anderen Form präsentiert wird als ein „normaler“ Arbeitsschritt. Zu diesem Zweck muss es möglich sein, bei der Modellierung zwischen präferierten Ausführungsreihenfolgen und Ausnahmepfaden zu unterscheiden.

3.1 Definition und Änderung von Ausführungsprioritäten

In diesem Abschnitt diskutieren wir Erweiterungen des ADEPT-Basismodells, die eine solche Unterscheidung ermöglichen.

Festlegung von Ausführungsprioritäten bei der WF-Modellierung

Um definieren zu können, ob eine Aktivität bei ihrer Aktivierung als normaler Arbeitsschritt oder als Ausnahmeschritt angeboten werden soll, führen wir als weiteres Modellelement statische *Ausführungsprioritäten* ein. Handelt es sich um einen Ausnahmeschritt, muss ihm bei der Modellierung die Priorität `EXCEPTIONAL` zugewiesen werden, andernfalls `REGULAR` (Voreinstellung). Die Ausführung von Ausnahmeschritten erfolgt nach denselben Regeln wie im Fall normaler Arbeitsschritte. Die Art und Weise, in der sie in Arbeitslisten angeboten werden, bleibt aber dem Anwendungsentwickler überlassen. Denkbar sind das Ein-/Ausblenden von Ausnahmeschritten, die Darstellung von Arbeitslisteneinträgen in unterschiedlichen Farben oder die Präsentation von normalen und ausnahmebedingten Arbeitsschritten in getrennten Arbeitslisten.

Die Verwendung von Ausführungsprioritäten erweist sich in Verbindung mit Parallelverzweigungen bei finaler Auswahl (AND-Split / XOR-Join) als nützlich. Hier kommt es zur gleichzeitigen Aktivierung paralleler Zweige, wobei für das Fortschreiten der WF-Kontrolle immer nur die Ausführung eines Teilzweiges notwendig ist. Bei kombinierter Anwendung mit Ausführungsprioritäten lassen sich somit präferierte Ausführungszweige von Ausnahmepfaden unterscheiden. Ein Beispiel zeigt Abb. 2.

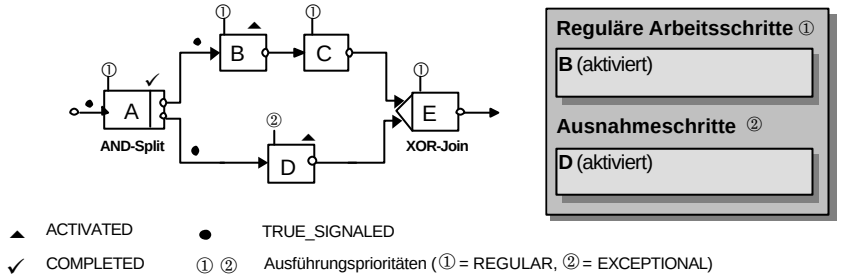

Abb. 2: Verwendung von Ausführungsprioritäten

Im dargestellten Ausführungsgraphen sind die Aktivitäten `B` und `D` gleichzeitig aktiviert. Infolge der zugeordneten Ausführungsprioritäten wird `B` den Benutzern als regulärer Arbeitsschritt und `D` als Ausnahmeschritt angeboten. Dementsprechend wird als Ausführungssequenz die Bearbeitung der Schritte `A`, `B`, `C` und `E` in der angegebenen Reihenfolge präferiert. Prinzipiell kann aber auch Aktivität `D` anstelle von `B` und `C` ausgeführt werden, etwa wenn eine Ausnahme ihre Ausführung zwingend erfordert.

Änderung von Ausführungsprioritäten während der WF-Ausführung

Abhängig vom WF-Status muss es möglich sein, eine Aktivität das eine Mal als Ausnahmeschritt und das andere Mal als regulären Arbeitsschritt zu behandeln. Soll z. B. die Bearbeitung in Ausnahmefällen vorzeitig erfolgen können, im Normalfall aber erst nach

Abschluß bestimmter anderer Schritte, so stellt sie vor Beendigung dieser Schritte eine Abweichung dar, während sie danach regulären Charakter besitzt.

Solche Sachverhalte lassen sich mit Hilfe statischer Prioritäten allein nicht modellieren. Es muss deshalb möglich sein, die Ausführungspriorität einer Aktivität im Verlauf der WF-Ausführung zu ändern. Zu diesem Zweck führen wir *Priorisierungskanten* ein. Einer Priorisierungskante ist eine der beiden Prioritäten `REGULAR` oder `EXCEPTIONAL` zugeordnet. Bei erfolgreicher Beendigung ihrer Quellaktivität wird ihrer Zielaktivität die Kantenpriorität als Ausführungspriorität zugewiesen. Besitzt eine Aktivität `Y` z.B. die statische Ausführungspriorität `EXCEPTIONAL` und wird zur Laufzeit eine in `Y` einmündende Priorisierungskante mit Kantenpriorität `REGULAR` signalisiert (d.h. die Quellaktivität wurde erfolgreich beendet), so wird die Ausführungspriorität von `Y` ebenfalls auf `REGULAR` gesetzt. Das bedeutet, dass die Ausführung dieses Schrittes keine Abweichung mehr darstellt und `Y` deshalb wie ein normaler Arbeitsschritt behandelt wird.

Ein Beispiel zeigt Abb. 3. Hier wird die Aktivität `D` zuerst als Ausnahmeschritt (vgl. Abb. 3 a) und nach Signalisieren der Priorisierungskante $C \rightarrow D$ als regulärer Schritt angeboten (vgl. Abb. 3 b). Insgesamt erlauben es die vorgestellten Erweiterungen, präferierte Ausführungsreihenfolgen und Ausnahmepfade voneinander zu unterscheiden.

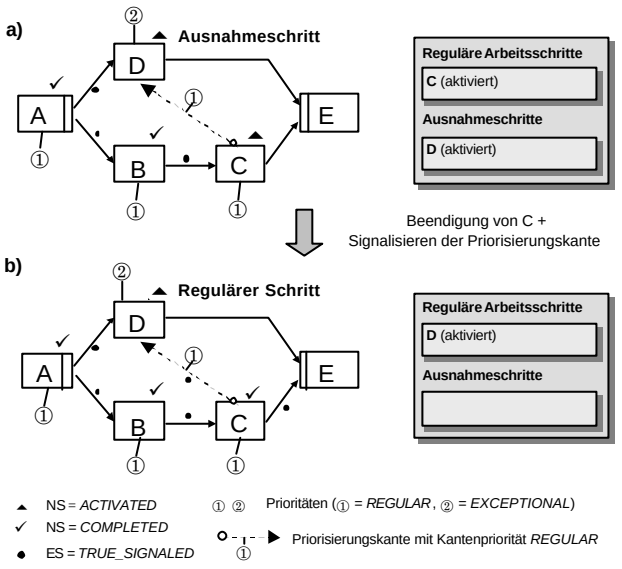

Abb. 3: Änderung der Ausführungspriorität eines Aktivitätenknotens

3.2 Modellierung von Vorwärtssprüngen

Wir zeigen nun, wie sich obige Modellelemente für die Modellierung planbarer Vorwärtssprünge (*Shortcuts*) nutzen lassen. Dabei unterscheiden wir Vorwärtssprünge mit und ohne Nachholen der übersprungenen Schritte. Auch Mischformen sind möglich, werden hier aus Platzgründen aber nicht behandelt. Voraussetzung für die Modellierung

eines Shortcut ist, dass die Zustände in denen er wählbar sein soll und die Aktivität(en), deren Bearbeitung im Ausnahmefall vorgezogen werden soll(en), a priori bekannt sind.

Aus Sicht des Modellierers erfolgt die Definition auf semantisch hoher Ebene, unter Verwendung einer speziellen Sprungkante (*Shortcut-Kante*). Diese wird intern in eine Darstellung des ADEPT-Basismodells übersetzt, wodurch eine präzise Ausführungssemantik resultiert. Ein Beispiel zeigt Abb. 4 a. Die Shortcut-Kante $A \rightarrow F$ spiegelt die Modelliersicht wider. Sie besitzt folgende Semantik: Nach erfolgreicher Beendigung der Quellaktivität n_s^{SC} (Knoten A im Bsp.) werden nicht nur die normalen, im Kontrollfluß nachfolgenden Aktivitäten von n_s^{SC} (B im Bsp.) aktiviert, sondern auch die Zielaktivität n_d^{SC} des Vorwärtssprungs (Knoten F im Bsp.). Diese wird solange wie ein Ausnahmeschritt behandelt, bis n_d^{SC} auf „normalen" Wege aktiviert wird (In unserem Bsp. trifft dies zu, nachdem E beendet wird.) Tritt dieser Fall ein und wurde n_d^{SC} zuvor nicht gestartet, wird die Aktivität in der Folge wieder wie ein normaler Arbeitsschritt behandelt. Wird n_d^{SC} dagegen vorzeitig gestartet, d.h. noch während der Status eines Ausnahmeschrittes besteht, müssen die dadurch übersprungenen Schritte (Knoten B, C oder D, E) gesondert behandelt werden. Hierzu werden nachfolgend zwei Varianten diskutiert.

Überspringen ohne Nachholen

Eine Möglichkeit zur Behandlung übersprungener Aktivitäten ist, auf ihre Bearbeitung komplett zu verzichten. Wählt ein berechtigter Akteur den Zielknoten eines Shortcut aus, während dieser den Status eines Ausnahmeschrittes besitzt, werden alle Aktivitäten zwischen dem Start- und Endknoten des Shortcut (im Bsp. B, C bzw. D und E) – abhängig von ihrem aktuellen Ausführungsstatus – kompensiert, abgebrochen oder deaktiviert. Anschließend wird mit der Ausführung des Sprungknotens (im Bsp. E) fortgefahren.

Ein Beispiel zeigt Abb. 4: Auf der linken Seite ist ein Shortcut dargestellt, wie er vom Modellierer definiert wird. Die rechte Seite zeigt seine Umsetzung in ADEPT. Sie basiert auf Ausführungsprioritäten und dem Konstrukt der Parallelverzweigung mit finaler Auswahl (AND-Split/XOR-Join). Ein Teilzweig enthält den Kontrollblock, dessen Aktivitäten bei Anwendung des Shortcut kompensiert, abgebrochen oder ausgelassen werden sollen. Der andere besteht aus einer vordefinierten, als Ausnahmeschritt gekennzeichneten Sprungaktivität <"Springe zu" + Sprungknoten>, die für berechtigte Akteure nach Beendigung des Shortcut-Quellknotens (A im Bsp.) und vor (regulärer) Aktivierung des Shortcut-Zielknotens (F im Bsp.) wählbar ist. Bei Auswahl der Sprungaktivität wird der obere Teilzweig beendet, woraufhin der untere Teilzweig abgebrochen und zurückgesetzt wird. Anschließend wird mit der Ausführung am Knoten F fortgefahren. Werden die Aktivitäten des unteren Zweiges dagegen auf normalem Wege beendet, d.h. ohne dass die Sprungaktivität zur Ausführung kommt, wird umgekehrt diese Ausnahmeaktivität deaktiviert. Die Ausführung von F stellt dann keine Abweichung mehr dar.

Bei der Shortcut-Definition werden verschiedene Überprüfungen vorgenommen, die eine korrekte Ausführung sicherstellen sollen. So wird z.B. geprüft, ob das zugehörige Datenfluss-Schema auch nach Umsetzung des Shortcut korrekt ist. Trifft dies bereits vor Hinzunahme der Shortcut-Kante zu, reduzieren sich die notwendigen Datenflussanalysen auf die Fragestellung, ob Lesezugriffe auf Prozessvariablen, die von Knoten des Sprungbereichs geschrieben werden, auch nach Auslassung dieser Knoten versorgt sind [Re00].

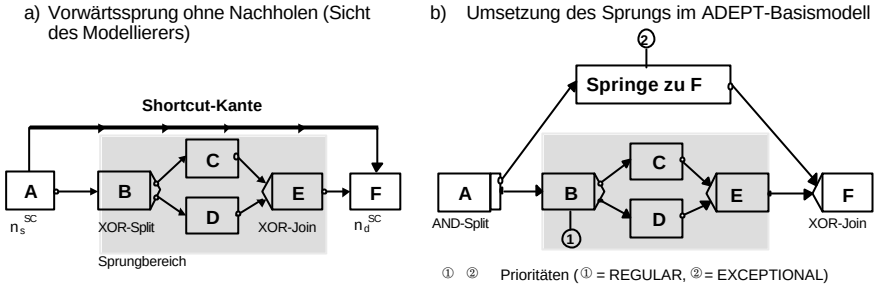

Abb. 4: Modellierung von Vorwärtssprüngen ohne Nachholen und Umsetzung in ADEPT

Überspringen mit Nachholen

Bei vorzeitiger Bearbeitung von Aktivitäten ist es nicht zwingend erforderlich, dass die dadurch übersprungenen Arbeitsschritte ausgelassen werden. Statt dessen ist es oftmals sinnvoll, diese parallel weiter zu bearbeiten. Für Aktivitäten des Sprungbereichs bedeutet das, dass die Effekte bereits beendeter Aktivitäten erhalten, begonnene Aktivitäten fortgeführt und noch nicht aktivierte Aktivitäten ausführbar bleiben sollen. Dieser Sachverhalt läßt sich in ADEPT ebenfalls modellieren. Der Modellierer muss dazu bei der Definition des Shortcut $n_s^{SC} \rightarrow n_d^{SC}$ einen weiteren Knoten n_n^{SC} festlegen, bis vor dessen Aktivierung die übersprungenen Schritte beendet bzw. nachgeholt werden sollen.

Ein Bsp. zeigt Abb. 5 a. Die Aktivitäten F und G sind nach Beendigung von A vorzeitig bearbeitbar, wobei die dadurch übersprungenen Aktivitäten (B, C bzw. D, E im Beispiel) vor Aktivierung von H beendet sein müssen. Die interne Umsetzung dieses Shortcut illustriert Abb. 5 b). Nach Beendigung von A wird B als regulärer Schritt und F als Ausnahmeschritt angeboten. Treten bei der WF-Ausführung keine Ausnahmen auf, d.h. werden jeweils nur Aktivitäten mit Priorität REGULAR bearbeitet, wird der untere Teilzweig der durch (A, H) gebildeten Verzweigung beendet, bevor F gestartet wird. F und G sind dann wieder als „normale“ Schritte ausführbar. Führt ein Benutzer dagegen F zuvor aus, liegt eine ausnahmebedingte Abweichung vor. In jedem Fall werden die Knoten des Sprungbereichs vor Aktivierung von H beendet. Formale Bedingungen für die korrekte Verwendung von Shortcuts und Graphtransformationsregeln finden sich in [Re00].

3.3 Diskussion und Zusammenfassung

Die vorgestellten Modellierungskonzepte können noch verallgemeinert werden. Beispielsweise sind auch Vorwärtssprünge abbildbar, bei denen die übersprungenen Schritte wahlweise nachgeholt oder ausgelassen werden. Konzeptuell ergeben sich hieraus keine neuen Fragestellungen, so dass wir an dieser Stelle auf eine Darstellung verzichten. Trivialerweise lassen sich durch die Verwendung von XOR-Verzweigungen auch automatische Vorwärtssprünge modellieren. Allerdings ist es in der Praxis nicht immer möglich, alle potentiellen Vorwärtssprünge a priori im WF-Modell zu berücksichtigen [DRK00]. Deshalb werden auch Mechanismen benötigt, die es Akteuren zur Laufzeit gestatten, Ad-hoc-Vorwärtssprünge durchzuführen (siehe [RD98]).

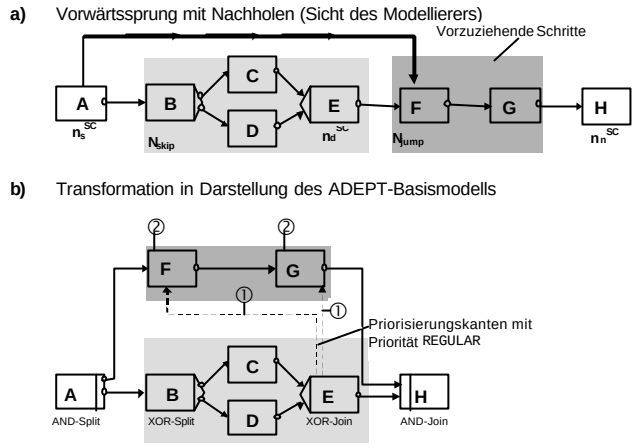

Abb. 5: Umsetzung von Vorwärtssprüngen mit Nachholen

4 Planbare Rückwärtssprünge

In ADEPT können auch verschiedene Arten von Rücksprüngen modelliert werden, von denen die wichtigsten nachfolgend betrachtet werden. Es lassen sich sowohl normale Schleifenrücksprünge als auch Fehlerrücksprünge, bei denen der WF in einen früheren Bearbeitungszustand rückgesetzt wird, modellieren. Dadurch können z.B. Aktivitätenfehlschläge (siehe Bsp. 2) automatisch behandelt werden.

Beispiel 2 (Scheitern einer Aktivitätenbearbeitung):
Wir beziehen uns auf Bsp. 1. Bei diesem Ablauf kann die Durchführung der medizinischen Intervention aus unterschiedlichen Gründen fehlschlagen, etwa aufgrund unterlassener Vorbereitungen oder fehlender Einwilligung des Patienten. Abhängig vom konkreten Grund des Scheiterns können unterschiedliche Ausnahmebehandlungen notwendig werden, wie der Abbruch des Ablaufs oder die Wiederholung bestimmter Schritte und die nochmalige Durchführung der Intervention.

4.1 Semantische Fehler und Fehlerrücksprungkanten

Zur Modellierung planbarer Rücksprünge führen wir Fehlercodes und -kanten ein. Fehlercodes definieren semantische Fehler einer Aktivität (im Bsp. „unterlassene Vorbereitung“ oder „fehlende Einwilligung“), die zur Modellierzeit bekannt sind und deren Auftreten zur Laufzeit zum Scheitern der Aktivitätenbearbeitung führt. Aufgabe des WF-Modellierers ist es, für solche a priori bekannten Ausnahmefälle geeignete Behandlungsformen festzulegen. Als mögliche Reaktionen unterstützt ADEPT die Wiederholung der Bearbeitung, die Ausführung alternativer Aktivitäten, das Überspringen der Aktivität, das Rücksetzen der Bearbeitung oder den kontrollierten Abbruch des WF.

Zur Definition von Rücksetzoperationen können *Fehlerkanten* verwendet werden. Eine Fehlerkante $n_{fail} \rightarrow n_{bwd}$ wird mit einem Fehlercode der Quellaktivität n_{fail} verknüpft.

Schlägt die Ausführung von n_{fail} fehl und wird der entsprechende Code gesetzt, so wird die WF-Bearbeitung unterbrochen und vor den Knoten n_{bwd} zurückgesetzt. Einige Beispiele zeigt Abb. 6. Die Fehlerkante `I→H` beschreibt einen Rücksetzoperation innerhalb eines Parallelzweiges, und die Kante `K→F` definiert einen Rücksprung in einen parallelen Teilzweig hinein. Einen Sonderfall beschreibt die Kante `B→Start`, bei deren Ausführung der WF vollständig rückgesetzt und abgebrochen wird.

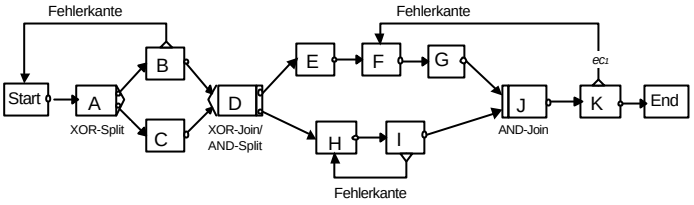

Abb. 6: Modellierung von Rücksprüngen durch Fehlerkanten

4.2 Automatische Fehlerrücksprünge

Automatische Fehlerrücksprünge werden mithilfe der vorgestellten Fehlercodes und -kanten modelliert. Bei ihrer Ausführung werden die Zustandsmarkierungen des WF-Graphen zurückgesetzt, so dass anschließend mit der Kontrolle am Rücksprungknoten fortgefahren werden kann. Des weiteren werden für abgeschlossene bzw. laufende Aktivitäten des Rücksetzbereichs interne Effekte (z.B. Schreiboperationen auf Prozessvariablen) zurückgenommen. Externe Auswirkungen (z.B. Änderungen von Anwendungsdaten) können durch die Ausführung von Kompensationsprogrammen rückgängig gemacht werden. ADEPT orientiert sich dabei am Sagas-Konzept [El92]. Ein Beispiel zeigt Abb. 7. Hier wird nach Fehlschlag der Aktivität `K` der Fehlercode *ec1* gesetzt. Dies führt zur Signalisierung der Fehlerkante `K → F`, was einen Rücksprung in den oberen Teilzweig der durch (D, J) definierten Parallelverzweigung bewirkt.

4.3 Benutzerinitiierte Fehlerrücksprünge

Für berechtigte Akteure muss es möglich sein, direkt in die Kontrolle der WF-Ausführung einzugreifen, indem sie den Ablauf unterbrechen und in einen früheren Bearbeitungszustand zurücksetzen. Solchen benutzerinitiierten Rücksprüngen liegen in der Mehrzahl der Fälle exogene Ursachen zugrunde, so dass der genaue Kontext ihres Auftretens a priori nicht oder nur ungefähr beschreibbar ist. Abhängig davon, ob eine Modellierung möglich ist oder nicht, unterscheiden wir zwischen planbaren Rücksprüngen und Ad-hoc-Rücksprüngen. Letztere klammern wir hier aus (siehe [Re00]).

Beispiel 3 (Benutzerinitiierte Rücksprünge)
Wir beziehen uns auf Bsp. 1. Läßt der aktuelle Zustand des Patienten die Untersuchung nicht mehr zu oder sollen zuvor getroffene Entscheidungen revidiert werden, muss der behandelnde Arzt die Ablaufkontrolle zurückerlangen können, solange die Untersuchung noch nicht stattgefunden hat. Dazu müssen ggf. laufende Tätigkeiten (z.B. Vorbereitungen des Patienten) unterbrochen oder zuvor abgeschlossene Schritte (z.B. Festlegung des Untersuchungstermins) storniert werden.

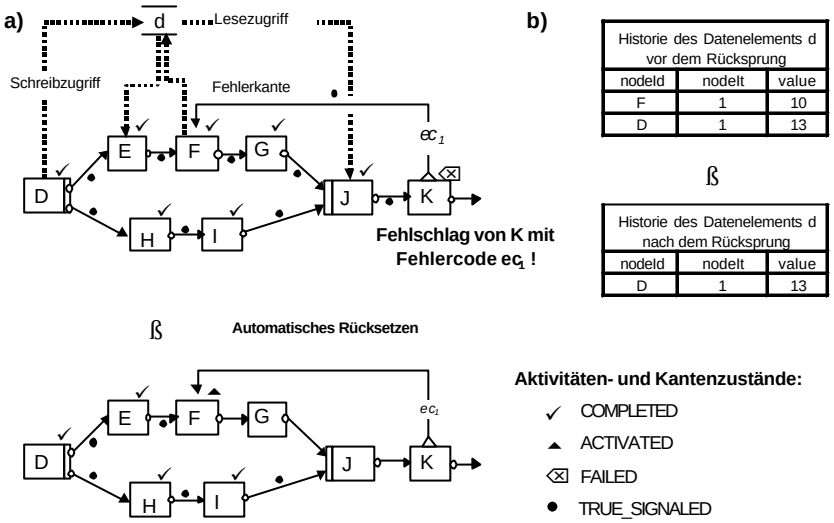

Abb. 7: Partielles Zurücksetzen nach Neumarkierung einer Fehlerkante durch a) Anpassung von Zustandsmarkierungen und b) Zurücknahme von Schreiboperationen auf Prozessvariablen

In ADEPT lassen sich Benutzerrücksprünge mittels spezieller Rücksprungkanten (*Regain-Control*) $n_s^{RC} \rightarrow n_d^{RC}$ modellieren: Ein Akteur kann zwischen Beendigung von n_d^{RC} und Aktivierung des im Ablauf nachfolgenden Knotens n_s^{RC} den WF in den Zustand vor Ausführung von n_d^{RC} zurücksetzen. Abb. 8 zeigt die Sicht des Modellierers. Der durch $C \rightarrow A$ beschriebene Benutzerrücksprung ist wählbar, nachdem A beendet und bevor C aktiviert wird. Bei seiner Selektion wird der WF in den Zustand vor Ausführung von A rückgesetzt.

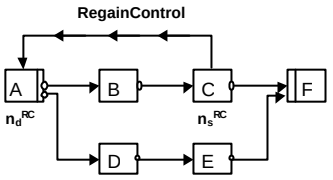

Abb. 8: RegainControl

4.4 Diskussion und Zusammenfassung

Auf Basis der vorgestellten Konzepte läßt sich ein breites Spektrum an Fehlerrücksprüngen modellieren. Allerdings ist es in der Praxis nicht immer möglich, alle Ausnahmen a priori zu modellieren. Das gilt auch für den Fall, dass der WF zur Laufzeit strukturell abgeändert wird und das Ziel des Rücksprungs eine Aktivität darstellt, die erst dynamisch während der WF-Ausführung eingefügt worden ist [Re00]. Aber selbst wenn diese Fälle nicht auftreten, ist eine Modellierung nicht immer möglich. Beispielsweise sind Rücksprünge in Teilzweige einer bedingten Verzweigung auf Modellebene nicht festlegbar, da a priori nicht bekannt ist, welcher Teilzweig zur Laufzeit ausgeführt wird. Aus diesen Gründen können berechtigte Akteure in ADEPT auch Ad-hoc-Rücksprünge im Ablauf vollziehen. Dazu kann der Benutzer aus einer Liste bereits beendeter oder laufender Arbeitsschritte einen Rücksprungknoten auswählen, vor dessen Ausführung der WF rückgesetzt werden soll. Details hierzu finden sich in [Re00].

5 Verwandte Arbeiten

Ein verbreiteter Formalismus zur WF-Modellierung sind Petri-Netze [Ob96]. Sie gehen oftmals davon aus, dass alle Ausnahmen bereits zur Modellierzeit bekannt sind [Ob92]. In der Mehrzahl der Fälle werden jedoch keine speziellen Modellierungskonzepte angeboten. Statt dessen müssen Ausnahmen mit denselben Sprachmitteln beschrieben werden wie Standardabläufe, was zu schwer verständlichen und unübersichtlichen Netzen führt [EKR95]. Neuere Entwicklungen greifen diese Kritikpunkte auf und bieten flexiblere Konzepte, etwa für das späte Modellieren von Subnetzen [HA97], die dynamische Anpassung von Netzmarkierungen [AM98], den Ad-hoc-Wechsel zwischen Netzkonfigurationen [BO98, Aa99] oder die dynamische Änderung der Netzstruktur [EKR95].

Im HieraStates-Projekt [Te98] wurde eine WF-Metamodell basierend auf State- und Activitycharts entwickelt. Es gestattet additive Änderungen (z.B. Hinzunahme neuer Zustände) zur Laufzeit. Darüber hinaus können planbare Vorwärtssprünge mit Hilfe spezieller Transitionen modelliert werden, die in verschiedenen Zuständen des Statechart aktivierbar sind. Rücksprünge und Rücksetzoperationen sind nicht modellierbar.

Im WfMS-Umfeld gibt es mehrere Arbeitsgruppen, die sich mit planbaren und nicht planbaren Abweichungen beschäftigen (z.B. [CFM99, EKR95, EL98, Jo99, We98]). MOKASSIN [Jo99] bietet dem Modellierer ein erweiterbares WF-Metamodell, mit dem spezielle Konstrukte für die Behandlung von Ausnahmen definiert werden können. In *Obligations* [Bo95] ergibt sich ein WF-Ausführungsgraph durch die Überlagerung mehrerer Prozess-Schablonen, die unterschiedliche Sichten des Ablaufs widerspiegeln. Eine Ausnahmebehandlung kann hier durch die Hinzunahme bzw. das Entfernen von Schablonen erfolgen. Ansätze wie AgentWork [MR99] erlauben es, Ausnahmen auf einer separaten Ebene durch einen Menge von Regeln zu beschreiben.

Eine detaillierte Diskussion dieser und weiterer Ansätze zur Behandlung von Ausnahmen in WfMS findet sich in [Re00]. Von ADEPT aufgegriffene, in obigen Ansätzen nicht behandelte Fragestellungen betreffen den Erhalt der Konsistenz von Zustandsmarkierungen bei Sprüngen, die Modellierung planbarer Sprünge, die korrekte Behandlung von Sprüngen in Verbindung mit Verzweigungen und Schleifen sowie die Unterstützung von Sprüngen unterschiedlicher Semantik (z.B. Vorwärtssprünge mit/ohne Nachholen, Schleifen-/Fehlerrücksprünge, automatische/spontane Sprünge).

6 Zusammenfassung

In diesem Beitrag haben wir Konzepte zur Modellierung planbarer Abweichungen in WfMS vorgestellt. Der Modellierer kann solche Abweichungen in einer abstrakten und verständlichen Notation formulieren. Durch ihre Übersetzung in einen begrenzten Satz einfacher Modellelemente erzielen wir eine effiziente und korrekte Ausführbarkeit. Des weiteren resultiert aus der getrennten Beschreibung von Standardablauf und Ausnahmen eine verbesserte Strukturierung der Modelle. Wir haben gezeigt, wie sich durch die Modellierung von Abweichungen ein flexibles Ausführungsverhalten erzielen läßt. Berechtigte Akteure können in die WF-Kontrolle eingreifen, indem sie zukünftige Arbeitsschritte vorzeitig ausführen oder den WF in frühere Bearbeitungszustände zurücksetzen.

Auch automatische Rücksprünge sind modellierbar. Die vorgestellten Konzepte bilden einen wichtigen Beitrag zur Erhöhung der A-priori-Flexibilität in WfMS und lassen sich prinzipiell auch auf andere WF-Beschreibungsformalismen anwenden.

Literaturverzeichnis

[Aa99] W. van der Aalst: Generic Workflow Models: How to Handle Dynamic Change and Capture Management Information? Proc. CoopIS'99, Edinburgh, 1999, S. 115–126

[AM98] A. Agostini, G. de Michelis: Simple Workflow Models. Proc. Workshop on Workflow Management, Lissabon, 1998, S. 146–163

[Bo95] D. Bogia: Supporting Flexible, Extensible Task Descriptions In and Among Tasks. Dissertation, University of Urbana, Illinois, 1995

[BO98] E. Badouel, J. Oliver: Reconfigurable Nets, a Class of High Level Petri Nets Supporting Dynamic Changes. Workshop Workflow Management, Lissabon, 1998, S. 129–145

[CFM99] F. Casati, M. Fugini, I. Mirbel: An Environment for Designing Exceptions in Workflows. Information Systems, 24(3):255-73, 1999

[DRK00] P. Dadam, M. Reichert, K. Kuhn: Clinical Workflows - The Killer Application for Process-oriented Information Systems? Proc. BIS'2000, Posen, 2000, S. 36–59

[EKR95] C.A. Ellis, K. Keddara, G. Rozenberg: Dynamic Change within Workflow Systems. Proc. Conf. Org. Computing Systems, Milpitas, S. 10–21, 1995

[El92] A.K. Elmargarmid (Hrsg.): Database Transaction Models for Advanced Applications. Morgan Kaufmann Publ., 1992

[EL98] J. Eder, W. Liebhart: Contributions to Exception Handling in Workflow-Management. Proc. EDBT-Workshop on Worklfow Management Systems, Valencia, 1998, S. 3-10

[Ha97] J. Hagemeyer, T. Hermann, K. Just-Hahn, R. Striemer: Flexibilität bei Workflow-Management-Systemen. Proc. Software-Ergonomie'97, Dresden, 1997, S. 179–190

[Jo99] G. Joeris: Defining Flexible Workflow Execution Behaviors. Proc. Workshop on Enterprise-wide and Cross-Enterprise Workflow-Mgmt., Paderborn, 1999, S. 49–55

[LR00] F. Leymann, D. Roller: Production Workflow. Prentice Hall, 2000.

[MR99] R. Müller, E. Rahm: Rule-Based Dynamic Modification of Workflows in a Medical Domain. Proc. BTW '99, Freiburg, 1999, S. 429-448

[Ob92] A. Oberweis: Spezifikation von Mechanismen zur Ausnahmebehandlung mit Petri-Netzen. Automatisierungstechnik - at, 40(1):21-30, 1992

[Ob96] A. Oberweis: Modellierung und Ausführung von Workflows mit Petri-Netzen, Teubner, 1996

[RD98] M. Reichert, P. Dadam: $ADEPT_{flex}$ – Supporting Dynamic Changes of Workflows Without Losing Control. Journal of Intelligent Information Systems, 10(2):93-129, 1998

[Re00] M. Reichert: Dynamische Ablaufänderungen in Workflow-Management-Systemen. Dissertation, Universität Ulm, Mai 2000

[SM95] D. Strong, S. Miller: Exceptions and Exception Handling in Computerized Information Processes, ACM ToIS, 13(2):206-33, 1995

[Te98] G. Teege: Flexible Workflows: Mitgestaltung durch die Ausführenden. Proc. Workshop Flexibilität und Kooperation in Workflow-Management-Systemen, 1998, S. 13–21.

[We98] M. Weske: Flexible Modeling and Execution of Workflow Activities. Proc. 31st Hawaii Int'l Conf. on System Sciences, Software Technology Track, 1998, S. 713–722