

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) bieten eine vielversprechende Technologie zur Realisierung prozessorientierter Anwendungen. Allerdings erlauben es heutige WfMS nicht, die betrieblichen Prozesse mit all ihren Sonderfällen und Ausnahmen adäquat abzubilden. Im ADEPT-Projekt haben wir hierfür 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 zu definieren, zum anderen können nicht vorhersehbare Abweichungen auch dynamisch zur Laufzeit erfolgen. Dieser Beitrag konzentriert sich auf den erstgenannten Aspekt. Er zeigt auf, wie sich ausnahmsbedingte Abweichungen in WfMS sinnvoll vormodellieren lassen, welche Anforderungen 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]. Eine charakteristische Eigenschaft vieler WfMS ist die Trennung von Prozesslogik und Anwendungscode [LR00]. Das bedeutet, dass die Ablauflogik der Arbeitsprozesse (engl. *Workflow*; kurz: WF) dem WfMS explizit durch die Modellierung bekannt gemacht und nicht im Programmcode „versteckt“ wird. Diese Trennung bietet mehrere Vorteile: Die Durchführung der Ablaufsteuerung durch das WfMS und die Bereitstellung entsprechender Systemdienste vereinfachen die Anwendungsentwicklung erheblich. Durch die explizite Beschreibung der Prozesslogik kann zudem das zukünftige Systemverhalten vorab evaluiert werden, wodurch sich Entwurfsfehler noch vor der Implementierung der eigentlichen Anwendungskomponenten entdecken lassen. Aus denselben Gründen können spätere Änderungen in den Geschäftsprozessen und daraus resultierende Anpassungen der Anwendungssysteme einfacher durchgeführt werden.

Trotz dieser Möglichkeiten sind WfMS in der betrieblichen Praxis nicht verbreitet. Ein Grund hierfür ist ihre mangelhafte Flexibilität [MR99, RD98, We98]. So sind die Ablaufmodelle vieler WfMS nicht ausdrucksmächtig genug, um die betriebliche Realität mit ihren üblichen Ausnahme- und Sonderfällen [SM95] adäquat beschreiben zu können. Des weiteren erlauben es heutige WfMS nicht oder nur sehr eingeschränkt, von dem einmal modellierten Ablauf eines Prozesses abzuweichen [Re00]. Solche Abweichungen sind aber in vielen Anwendungsbereichen unerlässlich [DRK00]. Ein WfMS muss deshalb zur Laufzeit ein flexibles WF-Ausführungsverhalten ermöglichen, andernfalls bleibt sein Einsatzspektrum auf einige wenige, starre Routineabläufe beschränkt.

Hierfür werden zum einen ausdrucksstarke Konzepte zur Modellierung planbarer Abweichungen benötigt, zum anderen muss das WfMS mächtige Mechanismen zur Behandlung nicht vorhersehbarer Ausnahmen bereitstellen. Zur besseren Differenzierung verwenden wir im Folgenden die Begriffe A-priori- und A-posteriori-Flexibilität [Jo99]. Erstere bezeichnet alle Maßnahmen, die der Entwickler zur Modellierzeit ergreifen kann, um das Ausführungsverhalten flexibel zu gestalten. Letztere fasst Funktionen zusammen, mit denen auf nicht planbare Ausnahmen reagiert werden kann. ADEPT unterstützt beide Facetten von Flexibilität. Zur Erhöhung der A-posteriori-Flexibilität gibt es fortschrittliche Konzepte für dynamische WF-Änderungen [RD98, Re00]. Sie gestatten es, die Struktur, den Status und die Attribute von WF-Instanzen dynamisch anzupassen. Berechtigten Akteuren stehen hierfür Handlungsmöglichkeiten offen, wie das spontane Einfügen oder Auslassen von WF-Schritten. Um dabei Fehler auszuschließen, werden bei der Durchführung von Ad-hoc-Änderungen relativ aufwendige Korrektheitsanalysen vorgenommen [Re00].

Dieser Beitrag behandelt Konzepte zur Verbesserung der A-priori-Flexibilität in WfMS. Motivation hierfür bilden langjährige Erfahrungen mit Geschäftsprozessen aus dem Krankenhausbereich [DRK00]. Hier hat es sich gezeigt, dass Abweichungen vom Standardablauf eines Arbeitsprozesses oftmals a priori bekannt sind und deshalb bei der Modellbildung berücksichtigt werden sollten. Dadurch läßt sich ein flexibles Ausführungsverhalten erzielen und es verringert sich die Häufigkeit der vergleichsweise teuren WF-Modifikationen zur Laufzeit. Notwendige Voraussetzung für die Vormodellierung von Ausnahmen sind die Vorhersehbarkeit ihres Kontexts und der zu ihrer Behandlung notwendigen Aktionen sowie ein ausdrucksstarker WF-Beschreibungsformalismus.

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 müssen präferierte Ausführungsreihenfolgen von ausnahmebedingten Abweichungen getrennt beschreibbar sein. Darüber hinaus darf unter der Einbeziehung von Ausnahmebehandlungen die Übersichtlichkeit der 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. Nur so können Anwender die Konsequenzen ihrer Interaktionen mit dem System verstehen und nachvollziehen.

Abschnitt 2 fasst wichtige Grundlagen des in ADEPT verwendeten WF-Metamodells zusammen, die für das weitere Verständnis dieses Beitrags nötig sind. In Abschnitt 3 nehmen wir Erweiterungen vor, mit denen 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 WF-Typ muss eine WF-Modell erstellt und im System hinterlegt werden. Ein solches WF-Modell beschreibt, welche Arbeitsschritte (sog. Aktivitäten) von welchen Benutzern in welcher Reihenfolge und unter Nutzung welcher Daten und Anwendungsprogramme bearbeitet werden sollen. Zur Modellierung der verschiedenen WF-Aspekte dient das ADEPT-Basismodell, das es gestattet, Prozesse möglichst natürlich und in einer für Anwender verständlichen Form zu beschreiben. Darüber hinaus ist die Überprüfung wichtiger Modelleigenschaften möglich, etwa im Hinblick auf Dynamikeigenschaften, Zeitbeschränkungen oder Datenflüssen.

Für die WF-Modellierung wird ein blockbasierter Beschreibungsansatz verfolgt, bei dem Sequenzen, Verzweigungen verschiedener Semantik (AND-AND; XOR-XOR und 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, werden zusätzliche Konstrukte angeboten, etwa Sync-Kanten zur Beschreibung verschiedener Arten von Wartet-Auf-Beziehungen zwischen Aktivitäten paralleler Zweige. Ein Beispiel zeigt Abb. 1.

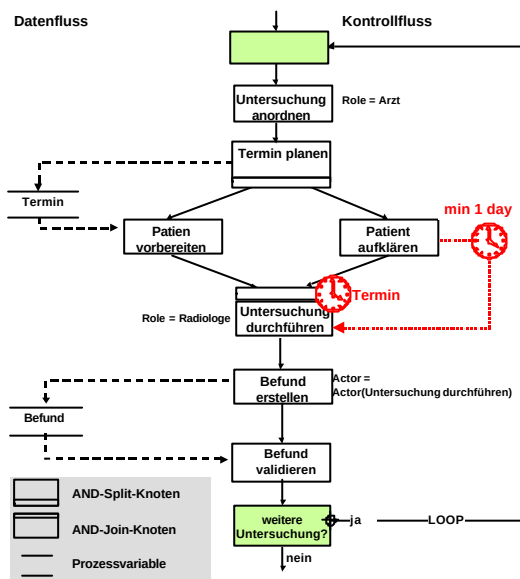


Abb. 1: Workflow-Modellierung in ADEPT

Arbeitslisten an, führen automatische Schritte selbständig durch und rufen die für die Ausführung einer Aktivität passenden Programme (mit den richtigen Daten) auf.

In den Abschnitten 3 und 4 diskutieren wir Erweiterungen des ADEPT-Basismodells. Sie gestatten die Vormodellierung bestimmter Arten von Abweichungen. Wir konzentrieren uns dabei auf planbare Vorwärts- und Rückwärtssprünge im WF-Graphen und gehen auf andere Mechanismen, wie das späte Modellieren eines (Sub-)WF, nicht ein.

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 Terminvereinba-

Um eine neue WF-Instanz zu erzeugen, müssen berechnigte Akteure das entsprechende WF-Modell aus dem Repository auswählen und vorgangsspezifische Daten festlegen. Nach Start der WF-Instanz wird diese über ihre komplette Lebensdauer hinweg, ggf. auch über einen längeren Zeitraum, vom WfMS kontrolliert. Dazu wird ein Ausführungsgraph verwaltet, dessen Knoten und Kanten bestimmte Zustandsmarkierungen besitzen. Es gibt wohldefinierte Regeln, die festlegen, unter welchen Zustandsmarkierungen eine Aktivität aktiviert werden darf und welche Folgemarkierungen sich im Anschluss an ihre Ausführung ergeben. Hierauf basierend steuern die Laufzeitkomponenten die WF-Ausführung, verwalten instanzspezifische Daten, bieten die zur Bearbeitung anstehenden Aktivitäten den zuständigen Akteuren in

rung und ohne die standardmäßigen Vorbereitungen notfallmäßig durchgeführt werden können. D.h. Anwender müssen diese Schritte überspringen können.

Vielfach sind Abweichungen vom Standardablauf eines Arbeitsprozesses planbar. Ist etwa a priori bekannt, dass die Bearbeitung einer Aktivität in Ausnahmefällen vorgezogen werden können muss, so sollte dieser Sachverhalt auch 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 Modellbildung zwischen präferierten Ausführungsreihenfolgen und Ausnahmepfaden zu unterscheiden. Im folgenden Abschnitt gehen wir auf die hierzu notwendigen Erweiterungen des ADEPT-Basismodells ein. Anschließend zeigen wir, wie sich darauf basierend Vorwärtssprünge modellieren lassen.

3.1 Definition und Änderung von Ausführungsprioritäten

In diesem Abschnitt diskutieren wir Erweiterungen des in Abschnitt 2 skizzierten ADEPT-Basismodells, die eine Unterscheidung zwischen präferierten Ausführungsreihenfolgen und Ausnahmepfaden ermöglichen sollen.

Festlegung von Ausführungsprioritäten bei der WF-Modellierung

Um definieren zu können, ob eine bestimmte Aktivität bei ihrer Aktivierung als normaler Arbeitsschritt oder als Ausnahmeschritt angeboten werden soll, führen wir als weiteres Modellelement statische *Ausführungsprioritäten* ein. Handelt es sich bei einer Aktivität um einen Ausnahmeschritt, muss ihr 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 etwa 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 Ausführungszweige, 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. Im dargestellten Ausführungsgraphen sind die beiden Aktivitäten B und D gleichzeitig aktiviert. Infolge der zugeordneten Ausführungsprioritäten wird B den Benutzern als regulärer Arbeitsschritt angeboten, während D als Ausnahmeschritt behandelt wird. Dementsprechend wird als Ausführungssequenz die Bearbeitung der Schritte A, B, C und E in der angegebenen Reihenfolge präferiert. Prinzipiell kann aber auch die Aktivität D anstelle von B und C ausgeführt werden, etwa wenn eine Ausnahme ihre Ausführung zwingend erforderlich macht.

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 Be-

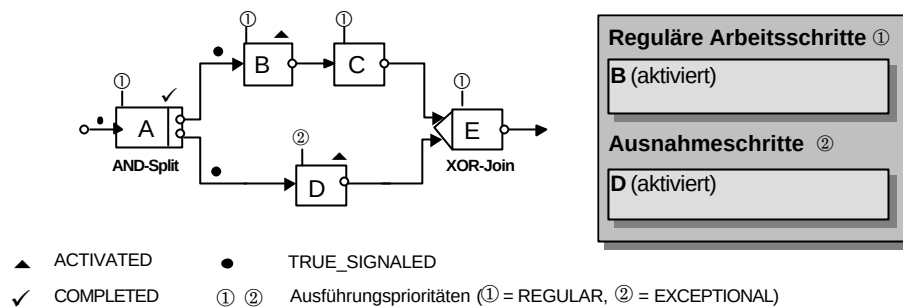


Abb. 2: Verwendung von Ausführungsprioritäten

arbeitung einer Aktivität in Ausnahmefällen vorzeitig erfolgen können, im Normalfall aber erst nach Abschluß bestimmter anderer Schritte, so stellt ihre Ausführung vor Beendigung dieser Schritte eine Abweichung dar, während sie danach regulären Charakter besitzt.

Solche Sachverhalte lassen sich mit Hilfe statischer Ausführungsprioritäten 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 (neue) Ausführungspriorität zugewiesen. Besitzt z.B. eine Aktivität Y 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 dementsprechend 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 → D als regulärer Schritt angeboten (vgl. Abb. 3 b). Insgesamt erlauben es die vorgestellten Erweiterungen, präferierte Ausführungsreihenfolgen und Ausnahmepfade voneinander zu unterscheiden.

3.2 Vormodellierung von Vorwärtssprüngen

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

Voraussetzung für die Modellierbarkeit 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 einer semantisch hohen Ebene, unter Verwendung einer speziellen Sprungkante (*Shortcut-Kante*). Diese wird intern in eine Darstellung des ADEPT-Basismodells übersetzt, wodurch eine präzise Ausführungssemantik resultiert.

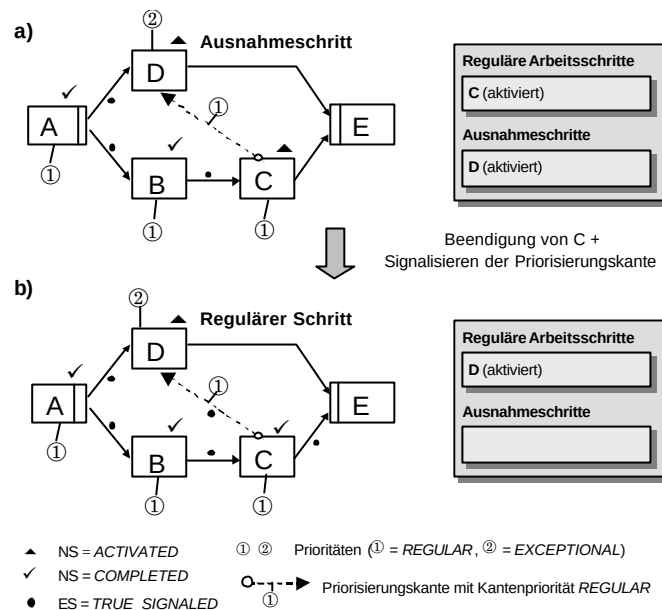


Abb. 3: Änderung der Ausführungspriorität eines Aktivitätenknotens

Ein Beispiel zeigt Abb. 4 a. Die Shortcut-Kante $A \rightarrow F$ spiegelt dabei die Sicht des Modellierers wider. Für ihre Aktivierung gilt folgende Regel: 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 allerdings wie ein Ausnahmeschritt behandelt – eine Sonderstellung die solange erhalten bleibt, 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 diese Aktivität in der Folge wieder wie ein normaler Arbeitsschritt behandelt. Wird die Aktivität n_d^{SC} dagegen vorzeitig gestartet, d.h. noch während sie den Status eines Ausnahmeschrittes besitzt, müssen die dadurch übersprungenen Schritte (Knoten B, C oder D, E) gesondert behandelt werden.

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 festgelegt 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

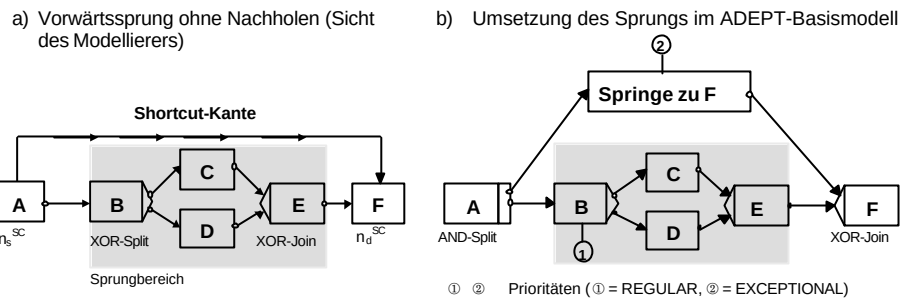


Abb. 4: Modellierung von Vorwärtssprüngen ohne Nachholen und Umsetzung in ADEPT

andere besteht aus einer vordefinierten, als Ausnahmeschritt gekennzeichneten Sprungaktivität <“Springe zu“ + Sprungknoten>, die für berechnete 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 im Sinne einer Ausnahme dar.

Bei der Shortcut-Umsetzung werden verschiedene Überprüfungen vorgenommen, die eine korrekte WF-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].

Überspringen mit Nachholen

Bei vorzeitiger Bearbeitung einer oder mehrerer Aktivitäten ist es nicht zwingend erforderlich, dass die dadurch übersprungenen Arbeitsschritte ausgelassen werden. Statt dessen ist es oftmals sinnvoll, die übersprungenen Aktivitäten 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 vormodellieren. 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 Beispiel hierzu zeigt Abb. 5 a. Hier sind die Aktivitäten F und G nach Beendigung von A vorzeitig bearbeitbar, wobei die dadurch evtl. übersprungenen Aktivitäten (B, C bzw. D, E im Beispiel) vor Aktivierung von H beendet sein müssen.

Die Umsetzung dieses Shortcut in das ADEPT-Basismodell illustriert Abb. 5 b). Nach Beendigung von A wird die Nachfolgeaktivität 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, so wird der untere Teilzweig der durch (A, H) gebildeten Verzweigung beendet, bevor F gestartet wird. Die Knoten F und G sind dann wieder als reguläre Aktivitäten ausführbar. Führt ein Benutzer die Aktivität F dagegen zuvor

aus, liegt eine Abweichung im Sinne einer Ausnahme vor. In jedem Fall werden die Knoten des Sprungbereichs vor Aktivierung von H beendet. – Formale Bedingungen für die korrekte Verwendung von Shortcuts und formale Graphtransmutationsregeln finden sich in [Re00].

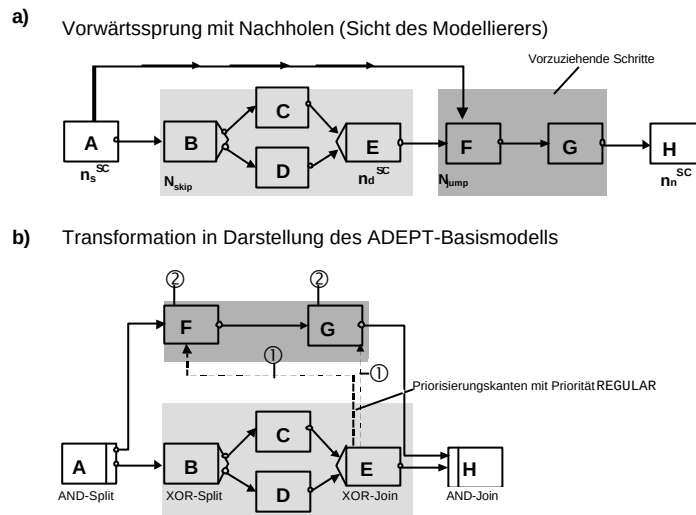


Abb. 5: Umsetzung von Vorwärtssprüngen mit Nachholen

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 auch automatische Vorwärtssprünge, durch Verwendung von XOR-Verzweigungen mit geeigneten Verzweigungsprädikaten, modellieren. Durch die Verwendung der vorgestellten Modellierungskonzepte lässt sich der Aufwand für Ad-hoc-Sprünge zur Laufzeit reduzieren. Allerdings ist es in der Praxis nicht immer möglich, alle potentiellen Vorwärtssprünge a priori im WF-Modell zu berücksichtigen [DRK00]. Zusätzlich zu den vorgestellten Konzepten werden deshalb Mechanismen benötigt, die es berechtigten Akteuren zur Laufzeit eines WF gestattet, Ad-hoc-Vorwärtssprünge durchzuführen (siehe [RD98]).

4 Planbare Rückwärtssprünge

Neben Vorwärtssprüngen können in ADEPT auch verschiedene Arten von Rücksprüngen modelliert werden. Es lassen sich sowohl normale Schleifenrücksprünge als auch Fehler-rücksprünge, bei denen der WF in einen früheren Bearbeitungszustand rückgesetzt wird, modellieren. Dadurch lassen sich z.B. Aktivitätenfehlschläge (siehe Bsp. 2) automatisch behandeln oder zuvor getroffene Verzweigungsentscheidungen revidieren.

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 fehlen-

der 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 Vormodellierung planbarer Rücksprünge führen wir Fehlercodes und Fehlerkanten 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 Behandlung semantischer Fehler können *Fehlerkanten* verwendet werden. Eine Fehlerkante $n_{fail} \rightarrow n_{bwd}$ wird mit einem Fehlercode der Quellaktivität n_{fail} verknüpft. Schlägt bei der WF-Ausführung die Bearbeitung von n_{fail} fehl und wird als Fehlerursache der entsprechende Code gesetzt, so wird die WF-Bearbeitung (teilweise) unterbrochen und vor den Zielknoten n_{bwd} der Fehlerkante zurückgesetzt. Einige Beispiele zeigt Abb. 6.

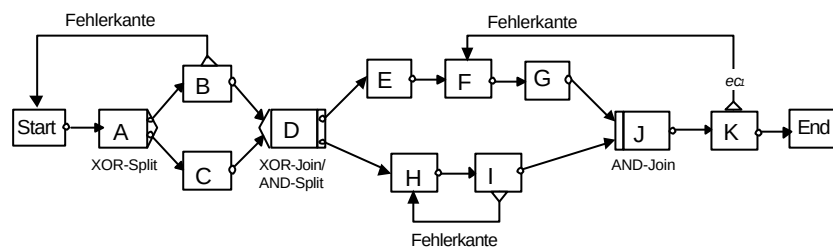


Abb. 6: Modellierung von Rücksprüngen durch Fehlerkanten

Die Fehlerkante $I \rightarrow H$ beschreibt einen Rücksetzoperation innerhalb eines parallelen Ausführungszweiges, und die Kante $K \rightarrow F$ definiert einen Rücksprung in den Teilzweig einer Parallelverzweigung hinein. Einen Sonderfall beschreibt die Kante $B \rightarrow \text{Start}$, bei deren Ausführung der WF vollständig zurückgesetzt und damit abgebrochen wird.

4.2 Automatische Fehlerrücksprünge

Automatische Fehlerrücksprünge werden direkt mit den vorgestellten Fehlercodes und -kanten modelliert. Bei ihrer Ausführung werden die Zustandsmarkierungen des WF-Graphen zurückgesetzt, so dass anschließend mit der Kontrolle an gewünschten 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 an dem Sagas-Konzept [El92].

Ein Beispiel zeigt Abb. 7. In dem dargestellten Ausführungsgraphen wird nach Fehlschlag der Aktivität K der Fehlercode *ec1* gesetzt. Dies führt zur Signalisierung der Fehlerkante

$K \rightarrow F$, was einen Rücksprung in den oberen Teilzweig der durch (D, J) definierten Parallelverzweigung bewirkt. Der Status des unteren Teilzweigs bleibt davon unbeeinträchtigt.

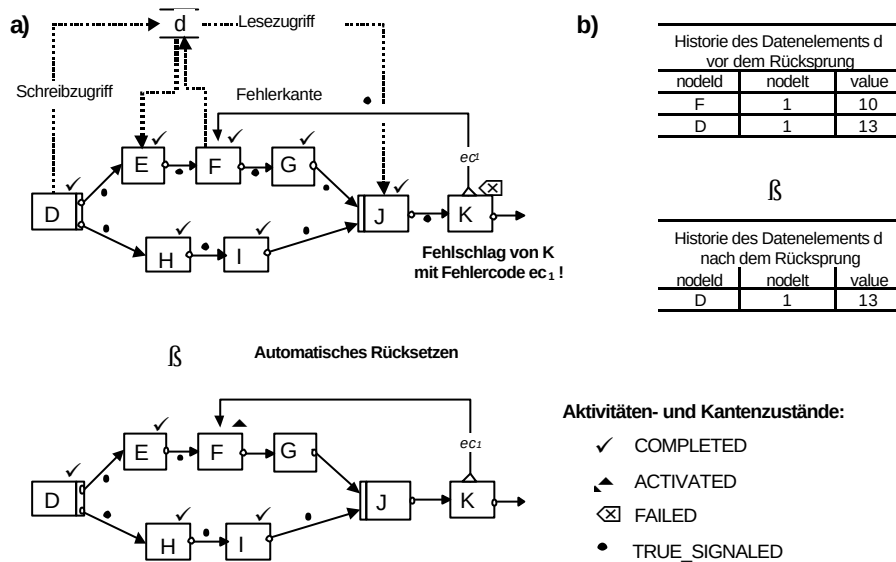


Abb. 7: Partielles Zurücksetzen nach Neumarkierung einer Fehlerkante durch a) Anpassung von Zustandsmarkierungen und b) Zurücknahme von Schreiboperationen auf Prozessvariablen

4.3 Benutzerinitiierte Fehlerrücksprünge

Für berechnigte Akteure muss es auch möglich sein, direkt in die Kontrolle der WF-Ausführung einzugreifen, indem sie den Ablauf unterbrechen und in einen früheren Bearbeitungsstand 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 Vormodellierung möglich ist oder nicht, unterscheiden wir zwischen planbaren Benutzerrücksprüngen und Ad-hoc-Rücksprüngen. Letztere klammern wir in diesem Beitrag aus (siehe hierzu [Re00]).

Beispiel 3 (Benutzerinitiierte Rücksprünge)

Wir beziehen uns wieder auf Bsp. 1. Lässt der aktuelle Zustand des Patienten die Untersuchung nicht mehr zu oder sollen zuvor getroffene Entscheidungen revidiert werden, muss ein behandelnder Arzt die Kontrolle über den Ablauf 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.

In ADEPT lassen sich solche Benutzerrücksprünge mittels einer speziellen Rücksprungkante (*RegainControl*) $n_s^{RC} \rightarrow n_d^{RC}$ modellieren: Ein Akteur kann zwischen der Beendigung von n_d^{RC} und der Aktivierung des im Ablauf nachfolgenden Knotens n_s^{RC} , den WF in den Zustand vor Ausführung von n_d^{RC} zurückzusetzen. Abb. 8 a) zeigt die Sicht des Modellierers. Die *RegainControl*-Kante $C \rightarrow A$ beschreibt einen Benutzerrücksprung. Er ist wählbar, nachdem A beendet und bevor C aktiviert wird. Bei seiner Selektion wird der WF in den Zustand zu-

rückgesetzt, der vor Ausführung von A gültig war. Intern wird dieses Konstrukt in eine Darstellung des ADEPT-Basismodells transformiert (Abb. 8 b), so dass eine eindeutige Ausführungssemantik resultiert. In Abb. 8b) werden nach Beendigung von A die Aktivitäten B und D als reguläre Schritte und die Rücksprungaktivität RC als Ausnahmeschritt aktiviert. Wird die WF-Instanz auf normalem Wege ausgeführt, wird nach Beendigung von B der Knoten n_3 (Nullaktivität ohne assoziierte Aktion) aktiviert und sofort wieder beendet, was die Deaktivierung der Rücksprungaktivität zur Folge hat. Startet dagegen ein Akteur RC bevor C aktiviert wird, bewirkt dies ein Rücksetzen des WF in den Zustand vor Ausführung von A.

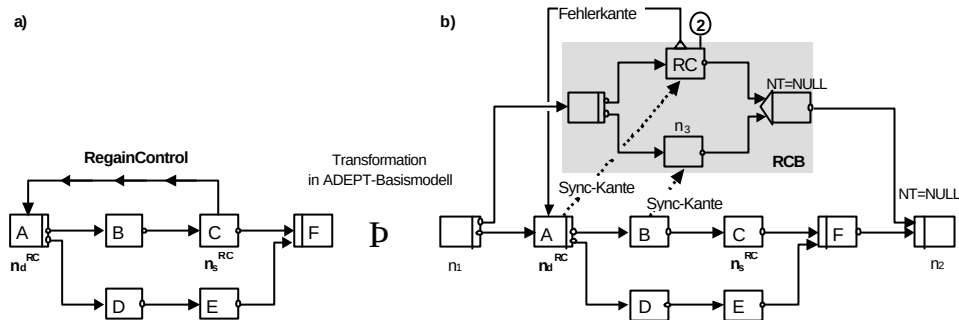


Abb. 8: a) RegainControl und b) Umsetzung in das ADEPT-Basismodell

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 Ausnahmesituationen vorherzusehen und modellseitig abzubilden. 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 Vormodellierung nicht immer realisierbar. Beispielsweise können Rücksprünge in Teilzweige einer bedingten Verzweigung auf Modellebene nicht festgelegt werden, da a priori nicht bekannt ist, welcher Teilzweig zur Laufzeit ausgeführt wird. Aus diesen Gründen können berechnete Akteure in ADEPT auch Ad-hoc-Rücksprünge im Ablauf realisieren. Dazu kann der Benutzer aus einer Liste bereits beendeter oder laufender Arbeitsschritte einen Rücksprungknoten auswählen, vor dessen Ausführung der WF rückgesetzt werden soll. Dabei kann es sich auch um eine Aktivitäteninstanz handeln, die in einer früheren Iteration einer Schleife ausgeführt wurde. Für diesen Fall müssen ggf. mehrere Schleifeniterationen zurückgenommen werden (siehe [Re00]).

5 Verwandte Arbeiten

Ein verbreiteter Formalismus für die WF-Modellierung sind Petri-Netze [Ob96]. Petri-Netz-Ansätze gehen oftmals davon aus, dass alle relevanten Ausnahmen bereits zur Modellierungszeit bekannt sind. Im Unterschied zu ADEPT werden jedoch keine speziellen Konzepte für die Modellierung planbarer Abweichungen angeboten. Statt dessen müssen diese 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 Netzausschnitten [BFG93, HA97], die dynamische Anpassung von Netzmarkierungen [AM98, EM97], den

dynamischen Wechsel zwischen vormodellierten 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 WF-Änderungen (z.B. Hinzunahme neuer Zustände) zur Laufzeit. Darüber hinaus können planbare Vorwärtssprünge mit Hilfe spezieller Transitionen vormodelliert werden, die in verschiedenen Zuständen des Statechart aktivierbar sind. Rücksprünge und Rücksetzoperationen sind in HieraStates nicht modellierbar.

Im WF-Umfeld gibt es mehrere Arbeitsgruppen, die sich mit planbaren und nicht planbaren Abweichungen in WfMS beschäftigen (z.B. [CFM99, Cu98, EKR95, EL98, HSS96, Jo99, We98]). MOKASSIN [Jo99] bietet dem Modellierer ein erweiterbares WF-Metamodell, bei dessen Verwendung 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. Andere Ansätze, wie AgentWork [MR99], erlauben es, Ausnahmen auf einer separaten Ebene durch eine Menge von Regeln zu beschreiben. DYNAMITE [He97] schließlich verwendet Graphgrammatiken, um die dynamische Evolution von Aufgabennetzen (inkl. Fehlerrücksprünge) kontrollieren zu können. Eine detaillierte Diskussion dieser und weiterer Ansätze findet sich in [Re00].

Von ADEPT aufgegriffene, in obigen Ansätzen nicht behandelte Fragestellungen, betreffen u.a. den Erhalt der Konsistenz von Zustandsmarkierungen, die Vormodellierung planbarer Sprünge, das Rücksetzen der WF-Kontrolle (unter Wahrung von Integritätsaspekten), 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

Die Ablaufmodelle heutiger WfMS sind nicht ausdrucksstark und flexibel genug, um die betriebliche Realität mit ihren Ausnahmen und ihrer Variabilität angemessen abbilden zu können. In diesem Beitrag haben wir Konzepte zur Vormodellierung von Abweichungen in WfMS vorgestellt. ADEPT erlaubt es dem Modellierer, Abweichungen vom Standardablauf in einer abstrakten und verständlichen Notation zu 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, ebenso wie das Überspringen von Aktivitäten. Die vorgestellten Konzepte bilden einen wichtigen Beitrag zur Erhöhung der A-priori-Flexibilität in WfMS, und lassen sich prinzipiell auch auf andere WF-Beschreibungsformalismen ausdehnen. Allerdings ist die vollständige Berücksichtigung von Ausnahmen auf Modellebene im allgemeinen nicht möglich, da nicht alle Ausnahmen vorhersehbar und damit planbar sind. Es werden deshalb zusätzliche Mechanismen benötigt, die dynamische WF-Änderungen zur Laufzeit gestatten. Solche Ad-hoc-Änderungen werden in ADEPT ebenfalls unterstützt.

Literaturverzeichnis

- [Aa99] W. an 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
- [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
- [BFG93] S. Bandinelli, A. Fuggetta, C. Ghezzi: Software Process Model Evolution in the SPADE Environment. IEEE ToSE, 19(12):1128-44, 1993
- [Bo95] D. Bogia: Supporting Flexible, Extensible Task Descriptions In and Among Tasks. Dissertation, University of Urbana, Illinois, 1995
- [CFM99] F. Casati, M. Fugini, I. Mirbel: An Environment for Designing Exceptions in Workflows. Information Systems, 24(3):255-73, 1999
- [Cu98] G. Cugola: Tolerating Deviations in Process Support Systems via Flexible Enactment of Process Models. IEEE ToSE, 24(11):982-1001, 1998
- [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
- [EL98] J. Eder, W. Liebhart: Contributions to Exception Handling in Workflow-Management. Proc. EDBT-Workshop on Workflow Management Systems, Valencia, 1998, S. 3-10
- [EKR95] C.A. Ellis, K. Keddera, G. Rozenberg: Dynamic Change within Workflow Systems. Proc. Conf. Org. Computing Systems, Milpitas, S. 10–21, 1995
- [El92] A.K. Elmagarmid (Hrsg.): Database Transaction Models for Advanced Applications. Morgan Kaufmann Publ., 1992
- [EM97] C.A. Ellis, C. Maltzahn: The Chautauqua Workflow System. Proc. 30th Hawaii Int'l Conf. on System Sciences, Maui, 1997
- [Ha97] J. Hagemeyer, T. Hermann, K. Just-Hahn, R. Striemer: Flexibilität bei Workflow-Management-Systemen. Proc. Software-Ergonomie'97, Dresden, 1997, S. 179–190
- [He97] P. Heimann, C. Krapp, B. Westfechtel, G. Joeris: Graph-based Software Process Management. Journal of Software and Knowledge Engineering, 7, 1997
- [HSS96] P. Heinl, H. Schuster, K. Stein: Behandlung von Ad-hoc-Workflows im MOBILE Workflow-Modell. Proc. STAK'96, München, 1996, S. 229–242
- [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
- [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