Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Ausarbeitung Seminar

Sven Stegelmeier

Ad-Hoc Workflow

Fakultät Technik und Informatik
Studiendepartment Informatik

Sven Stegelmeier

Thema der Ausarbeitung
Seminar

Ad-Hoc Workflow

Stichworte

adaptiv, flexbibel, Workflow, WFMS, Geschäftsprozess

Kurzzusammenfassung

In der sich heutzutage immer schneller ändernden Geschäftswelt, verlangen Unternehmen nach Möglichkeiten ihre Geschäftsprozesse zu steuern und zu koordinieren. Traditionelle Workflowmanagementsysteme (WFMS) sind mit ihren statischen Prozessdefinitionen nicht mehr in der Lage den heutigen Anforderungen gerecht zu werden. In diesem Bericht werden verschiedene Ansätze zur Realisierung von Ad-Hoc Workflows gezeigt.

Inhaltsverzeichnis

1 Einleitung **3**

2 Ad-Hoc Routing und Dynamische Ressourcenzuteilung **4**

3 Agentenbasiertes Workflowmanagement **6**

4 Ad-Hoc Strukturänderungen auf Basis von Petrinetzen **8**

5 Ad-Hoc Strukturänderungen durch partizipative Modellierung **10**

6 Freiheitsgrade und Strukturänderungen durch Meta-Modell **11**

7 Master Thesis **12**

8 Zusammenfassung **13**

1 Einleitung

Anfang der 90er Jahre führte die immer stärker werdende Globalisierung zu einen großen Konkurrenzdruck in den Unternehmen. Als Folge versuchten diese ihre Tätigkeiten zu optimieren. Geschäftsprozesse wurden erkannt und modelliert. Ein solches Modell diente als Diskussionsgrundlage für Experten, durch die eine Optimierung der Abläufe in einem Unternehmen erreicht werden sollte. Geschäftsprozesse sollten durch Rechner in Form von Workflowmanagementsystemen unterstützt werden. Dazu wurden diese in einer maschinenlesbaren Form, dem sog. Workflow, aufgeschrieben.

Ein Workflowmanagementsystem hatte nun die Aufgabe Instanzen von einem Workflowschema, welches einen modellierten Workflow darstellt, zu erstellen, auszuführen und zu verwalten. Diese neue Art der Ausführung von Unternehmenstätigkeiten sollte eine Ära einleiten, in der Faktoren wie Time-To-Market, Agilität und die Kosten deutlich verbessert werden würden. Allerdings sind heutzutage nur ein Bruchteil der über 1000 Workflowprodukte erfolgreich.

Der Grund für diese Entwicklung liegt in der sich immer schneller ändernden Geschäftswelt. Diese Änderungen haben starken Einfluss auf die auszuführenden Workflows. Passt der modellierte Workflow nicht mehr zur Realität müssen Workarounds gefahren und das Workflowmanagementsystem kurzzeitig verlassen werden. Irgendwann laufen Modell und Realität soweit auseinander, dass der Nutzen des Systems, die optimierte Ausführung des Prozesses in Bezug auf Geschwindigkeit und Steuerung, nicht mehr ausreichend ist um die hohen

Anschaffungskosten zu amortisieren. Zur Kompensation von Änderungen in der Geschäftswelt gibt es in der Regel zwei Möglichkeiten (vgl. Joeris (2000)). Die Erste ist die flexible Modellierung von Workflows, bei der Freiheitsgrade vorab definiert werden. Diese ist allerdings mit den heutigen Workflowmodellierungssprachen wie z.B. BPEL, BPML oder XPDL nicht möglich. Die Zweite Möglichkeit wäre die Anpassung des Modells während der Ausführung. Diese Möglichkeit wird allerdings von den heutigen Workflowmanagementsystemen nicht unterstützt.

Die beiden eben aufgezählten Möglichkeiten führen zu einer neuen Art des Workflows, dem sog. Ad-Hoc Workflow. Allerdings ist die Grenze zwischen den traditionellen und den Ad-Hoc Workflows nicht klar zu ziehen. In Huth und Nastansky wird ein Workflowkontinuum vorgestellt, bei dem feine Abstufungen zwischen den beiden Extremen festgestellt wurden. Im Folgenden werden nun Ansätze für die Realisierung von Ad-Hoc Workflowmanagementsystemen vorgestellt. Dieser Bericht schließt mit einem Ausblick auf eine Master-Thesis in diesem Bereich ab.

2 Ad-Hoc Routing und Dynamische Ressourcenzuteilung

Eine Möglichkeit die oben beschriebenen Probleme im Gebiet des Workflowmanagements anzugehen wurde in Eric Y.Shan und Shan (2004) vorgeschlagen. Hierbei wurde ein bestehendes Workflowmanagementsystem an drei Punkten, von denen hier zwei vorgestellt werden, so abgeändert das eine Möglichkeit besteht Änderungen in der Geschäftswelt für das System sichtbar zu machen. Die Änderungen wurden auf den folgenden Ebenen vorgenommen:

Auf Modellebene: Durch spezielle Knoten wird der Durchlauf durch den Workflowgraphen zur Laufzeit bestimmt.

Auf Ressourcenebene: Eine Ressource wird dynamisch zu einer Aufgabe zugeteilt.

Die erste Änderung erfolgte auf der Modellebene. Hier wird ein Workflow durch einen Workflowgraphen beschrieben. Die Knoten dieses Graphen sind dabei die zu erledigenden Aufgaben. Die von einem Knoten ausgehenden Kanten sind die verschiedenen Möglichkeiten, mit denen es im Verlauf des Workflows weitergehen kann. Zu jeder dieser Kanten gehören Bedingungen, die der entsprechende Knoten auswertet um die richtige Alternative zu bestimmen. Sie sind statisch in dem Knoten kodiert und ändern sich in der Regel nur durch eine

Modelländerung des Workflowschemas. Die Idee ist hier, diese Bedingungen zur Laufzeit neu festzulegen. Dadurch ändert sich der traditionelle Routing-Node zu einem Choice-Node. Dieser fragt zur Laufzeit eine Regelbasis ab, die die Änderungen in der Geschäftswelt reflektiert und ständig so aktualisiert wird, dass immer ein möglichst optimaler Pfad durch den Workflowgraphen genommen wird. Ein Beispiel wäre hier die Entscheidung ob der Transport einer Ware per Luft, See oder zu Lande erfolgen soll. Die optimalen Entscheidungen können durch die verschiedenen Anforderungen an die einzelnen Transportunternehmen variieren. Steigen z.B. schlagartig die Preise für Lufttransport auf Grund von Flugunfällen, so kann ad-hoc auf Land- oder Seetransport umgeschaltet werden.

Die zweite Änderung ist analog zur ersten allerdings auf einer tieferen Ebene. Ist durch einen Routing- oder einen Choice-Node die Auswahl für eine bestimmte Aufgabe getroffen worden, muss dieser eine Ressource zugewiesen werden. Dies sind z.B. Clientanwendungen, an denen Menschen arbeiten, aber auch voll automatisierte Ressourcen wie Roboter. Auch hier waren die Entscheidungen für die Zuweisung von Ressourcen zu Aufgaben bisher durch Rollenauflösungen statisch festgelegt. Analog zu den Choice-Nodes soll auch hier eine Regelbasis abgefragt werden, die die Änderungen in der Geschäftswelt reflektiert. Durch diese Maßnahme kann z.B. der Durchsatz einer Fertigungsstraße optimiert werden.

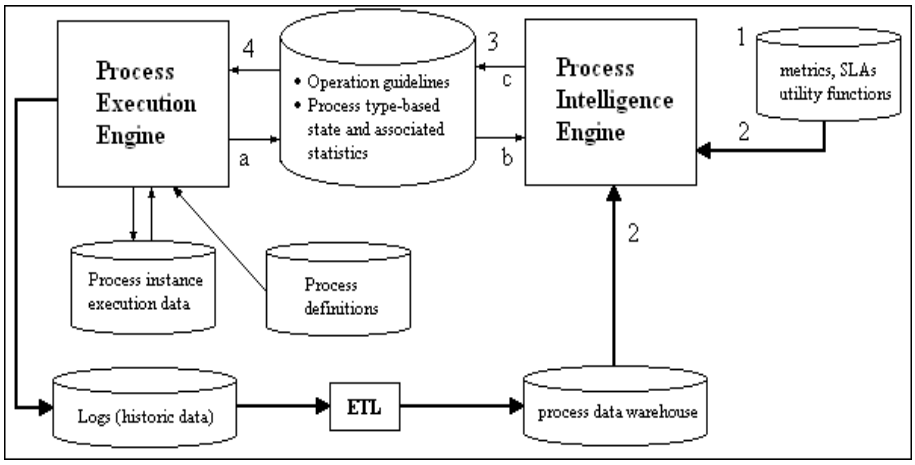

Abbildung 1: Architektur eines intelligenten WFMS

Die in Abbildung 1 dargestellte Architektur soll nun das Fundament für die nötige Entscheidungsgrundlage der oben beschriebenen Eingriffspunkte liefern. Sie besteht aus zwei Kreisläufen, die für die Aktualisierung der Wissensbasis verantwortlich sind. Der äußere Kreislauf (Punkte 1,2,3 und 4) extrahiert durch Logfile-Mining Daten aus dem Produktionsfluß von Workflows und speichert diese in einem Datawarehouse. Aus diesen Daten werden durch die Process Intelligence Engine zusammen mit vom Benutzer definierten Metriken und Richtlinien Regeln generiert. Diese Regeln werden dann für die Entscheidungen auf Modellebene

verwendet. Der innere Kreislauf (Punkte a,b und c) nimmt Statusinformationen von der Process Execution Engine und wertet diese in Bezug auf die Auslastung von Ressourcen aus. Dadurch können Regeln für die Entscheidung auf Ressourcenebene erstellt werden.

3 Agentenbasiertes Workflowmanagement

Bei diesem Ansatz steht die dezentrale Ausführung von Workflows im Vordergrund. Sie steht in einem klaren Gegensatz zu den traditionellen Workflowmanagementsystemen, die eine zentrale Stelle bilden, von wo aus alle Workflows gesteuert und verwaltet werden. Motiviert ist dieser Ansatz aus der heutigen Servicelandschaft im Internet. Es gibt eine Unmenge von verschiedenen Diensten, die ihren Standort wechseln oder gar ganz verschwinden. Oft sind diese Dienste synchron, wodurch eine Kommunikation über das Internet durch die unsicheren Verbindungen erschwert wird. Auf der einen Seite würde eine asynchrone Kommunikation durch Callback Mechanismen realisiert werden können. Diese sind allerdings gerade in Bezug auf Web Services erst in der Entwicklung. Auf der anderen Seite, und das ist die Philosophie dieses Ansatzes, kann eine asynchrone Kommunikation auch über einen *Mobilen Agenten* erledigt werden.

Dieser mobile Agent wandert von Dienst zu Dienst und erledigt dort Aufgaben. Ist eine Verbindung einmal unterbrochen, so wartet der Agent bis diese wieder besteht und wandert weiter. Nach Beendigung aller Aufgaben kehrt er zu seinem Ursprung zurück und der Workflow ist abgeschlossen.

Dieser Ansatz richtet sich also sehr stark auf dienstorientierte Umgebungen wobei Aufgaben in einem Workflow von Diensten erledigt und diese damit als Ressourcen angesehen werden. Im Folgenden wird die Architektur eines solchen verteilten Systems beschrieben.

Abbildung 2 zeigt die konzeptionelle Architektur das verteilten WFMS. Einzelne Knoten sind über das Internet verbunden, welches als Transportmittel für den mobilen Agenten dient. Damit so ein Ansatz funktionieren kann, benötigen alle teilnehmenden Dienste eine Infrastruktur für mobile Agenten. In Jie Meng wurde dafür das Voyager MA System der Firma ObjectSpace verwendet. Jeder Knoten verfügt neben dieser Middleware über seinen lokalen Dienst und eine Rule Engine. Letztere enthält Geschäftsregeln für diesen Dienst. Sie ermöglichen das Ableiten von Entscheidungen einerseits für den aufzurufenden Dienst bei der Ankunft des Agenten und andererseits für das Weiterleiten des Agenten zum nächsten Dienst. Z.B. wenn ein Agent bei einem Händler eine Bestellung macht, so wird dieser Agent einen speziellen Dienst angeboten bekommen falls er z.B. von der Regierung kommt. Nach

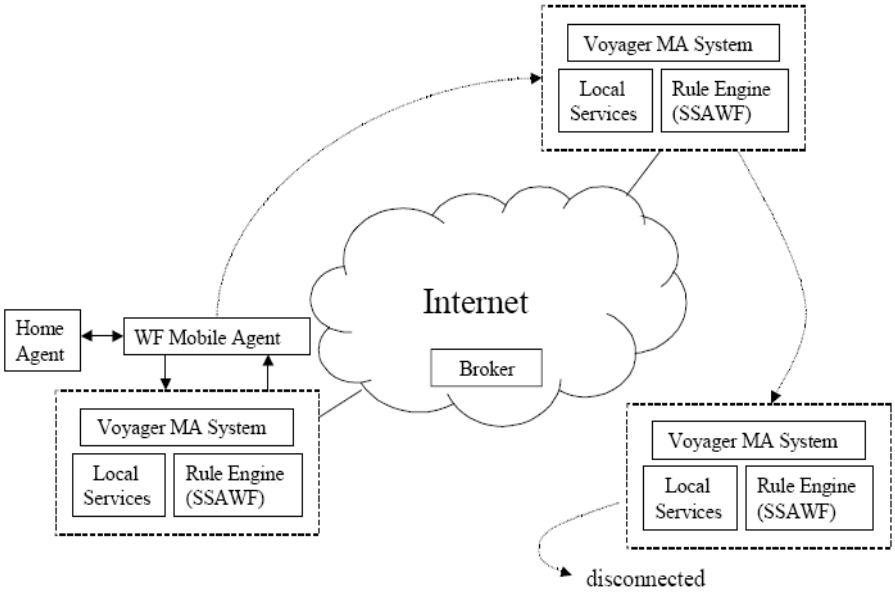

Abbildung 2: Architektur eines agentenbasierten Workflowmanagementsystems

der erfolgreichen Bestellung muss der Agent zu einem Transportunternehmen weitergeleitet werden um den Transport der Ware in Gang zu setzen. Dies ist die Dienstseite der Regeln und wird als SSAWF (Service Side Ad-Hoc Workflow) bezeichnet. Die andere Seite der Regeln ist die MASAWF (Mobile Agent Side Ad-Hoc Workflow). Diese sagt dem Agenten was er tun soll. Z.B. muss er wenn er bei einem Händler ein Bestellung machen soll im Lager nachschauen ob die benötigte Bestellmenge vorhanden ist.

Wenn der Agent von einem Dienst zum nächsten wandert, so muss er die konkreten Adressen der Rechner im Internet wissen. Damit diese vernünftig verwaltet werden können, kommt als zentrales Element dieser Architektur ein Broker ins Spiel, der symbolische Dienstnamen auf konkrete Adressen abbildet. Bekommt der Agent z.B. von den Regeln der SSAWF gesagt, dass er zu einem Lieferanten wandern soll, so fragt er den Broker nach Lieferantendiensten. Über seine internen Regeln (MASAWF) wählt er dann einen passenden aus.

Ein verteilter Ad-Hoc Workflow muss natürlich auch verwaltet werden können, damit der Initiator das Fortschreiten beobachten und vielleicht sogar bei bestimmten Entscheidungen eingreifen kann. Dies wird über einen Home Agenten erledigt. Zu jedem Workflow wird ein mobiler und ein Home Agent erstellt. Während der mobile Agent durch die Welt wandert steht er immer in Kontakt mit dem Home Agenten.

4 Ad-Hoc Strukturänderungen auf Basis von Petrinetzen

Eine weitere Methode Ad-Hoc Workflows zu unterstützen ist die Struktur des Workflows zur Laufzeit zu ändern. Hier entstehen dann ganz neue Pfade und Alternativen. Damit dies realisiert werden kann, muss die Modellierungssprache des WFMS mit entsprechenden Konstrukten ausgestattet werden. In Voorhoeve und van der Aalst (1997) wird dazu ein Ansatz gezeigt, der als Sprache Petrinetze verwendet.

Die grundsätzliche Annahme ist, dass es eine Bibliothek von Workflowschablonen gibt. Diese Schablonen bilden eine Hierarchie und sind Workflows auf einem hohen Abstraktionsniveau. Schablonen in den höheren Ebenen sind so stark abstrahiert, dass diese Workflows nicht ausgeführt werden können. Die unteren Ebenen sind mögliche konkrete Ausprägungen von abstrakten Aktionen aus den Top-Level Schablonen. Diese Ausprägungen können ausgeführt werden und sind Subworkflows ihrer Väter in der Hierarchie.

Das Ziel ist es einen Workflow anhand einer Schablone auszuwählen und seine konkreten Ausprägungen falls nötig ad-hoc abzuändern. Dazu werden drei wichtige Kategorien von Regeln gegeben, die im Folgenden kurz vorgestellt werden.

- Regeln für das Einfügen von Netzen

  Refinement: Wird verwendet um ein anderes Netz in ein bestehendes einzusetzen. Eine Transition im Originalnetz wird dabei mit dem einzufügenden Netz substituiert.

  Reduction: Ist das Komplement zum Refinement. Wenn zwei Knoten in einem Netz so in Beziehung stehen, dass die ausgehende Kante des einen die eingehende des anderen ist und dies gleichzeitig noch die einzige Kante für Aus- bzw. Eingang der beiden Knoten ist, so können diese beiden Knoten zu einem zusammengefasst werden.

  Extension: Ist die Inverse Operation zur Reduction. Sie erzeugt eine Situation, wie im vorigen Punkt beschrieben.

- Regeln für das Hinzufügen von einzelnen Stellen und Transitionen

  And Split: Fügt eine Stelle zwischen zwei Transitionen hinzu. Auf beiden Stellen müssen demnach Marken liegen, damit die nachfolgende Transition schalten kann.

  Or Split: Fügt eine Transition zwischen zwei Stellen hinzu. Demnach kann eine der beiden schalten. Dies modelliert ein XOR, was ein besserer Name für diese Regel sein müsste.

Iteration: Fügt einer Stelle eine Transition hinzu, die nur mit dieser Stelle verbunden ist. Sie kann daher beliebig oft schalten, solange eine Marke auf dieser Stelle liegt.

- Regeln für die Synchronisation von Aufgaben

  Einfache Synchronisation: Synchronisiert zwei Transitionen in einer festen Reihenfolge durch Verbinden der beiden Transitionen durch eine Stelle.

  Duale Synchronisation: Ersetzt zwei Transitionen durch eine. Dadurch sollen die beiden vorherigen Transitionen quasi simultan ausgeführt werden.

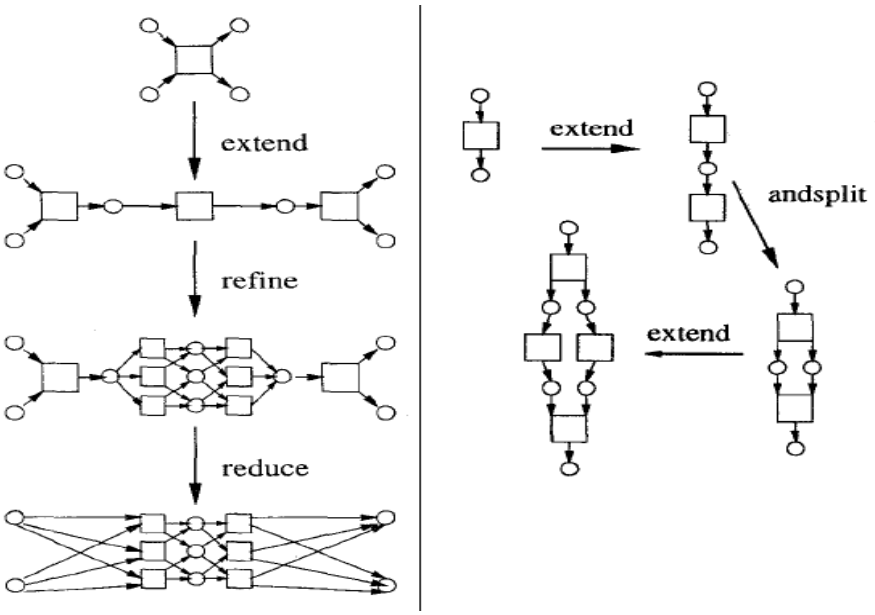

Abbildung 3: Beispiel für Regelbenutzung

Abbildung 3 zeigt ein Beispiel für die kombinierte Benutzung von den oben angesprochenen Regeln. Das Beispiel auf der linken Seite zeigt, wie durch Kombination der Regeln Extension, Refinement und Reduction ein Refinement für eine beliebige Anzahl von Transitionen gemacht werden kann. Das Beispiel auf der rechten Seite zeigt, wie durch Extension und And Split ein Task durch zwei parallele Tasks mit einem gemeinsamen Anfang und Ende ersetzt werden können.

5 Ad-Hoc Strukturänderungen durch partizipative Modellierung

Der in Huth und Nastansky präsentierte Ansatz zielt auf eine integrierte Büroumgebung zur Ausführung von Ad-Hoc Workflows ab. Ein traditionelles Workflowmanagementsystem soll durch eine zusätzliche Komponente erweitert werden, um Ad-Hoc Workflows unterstützen zu können. Die Grundannahme ist, dass Ad-Hoc Workflows eher in kleinen Teams in einer Abteilung ablaufen und nur selten mehrere Abteilungen oder sogar Unternehmen mit einbeziehen. Als Konsequenz dessen wurden fünf Änderungen an den Paradigmen des traditionellen Workflowmanagements vorgenommen um Ad-Hoc Workflows zu unterstützen. Drei dieser Änderungen werden im Folgenden kurz angesprochen und anschließend ein Überblick über die Implementierung gegeben.

Verschmelzung von Gestaltungs- und Laufzeitumgebung: In traditionellen Workflowmanagementsystemen sind die Gestaltungs- und Laufzeitebene strikt getrennt. D.h. ein Workflow wird modelliert und dann werden von dem Modell beliebig viele Instanzen gebildet und ausgeführt. Da bei einem Ad-Hoc Workflow die Struktur oft noch nicht bekannt ist, kann kein Modell vorab gebildet werden. Zusätzlich muss es gerade möglich sein, dass ein Workflow zur Laufzeit erst modelliert wird.

Verschmelzung von Modell und Instanz: Ein Ad-Hoc Workflow hat erst nach Abschluss ein vollständiges Modell. Das Speichern dieses Modells lohnt sich allerdings kaum, da der Prozess sehr vielen Änderungen unterworfen sein wird und nur eine geringe Wiederholungsfrequenz besitzt. Stattdessen werden Instanz und Modell zusammengefasst und gespeichert. Sollte so ein Prozess erneut ausgeführt werden, so kann die gespeicherte Instanz als Vorlage genommen und weiter verändert werden.

Partizipative Modellierung von Workflows: Im traditionellen WFM werden Workflowmodelle durch einen Modellierungsexperten erstellt. Dieser spricht meist nur mit den Prozessverantwortlichen und bekommt dort nur eine eingeschränkte Sicht über den zu modellierenden Prozess. Viel Wissen liegt vor Ort bei den beteiligten Mitarbeitern begraben und soll durch ein Miteinbeziehen in die Modellierung nutzbar gemacht werden.

Die oben kurz vorgestellten Änderungen am traditionellen Workflowmanagement wurden in dem GroupProcess System umgesetzt. Dieses System basiert auf der Programmiersprache Java, die die Interoperabilität sichern soll. Diese wird aufgrund der Zusammenlegung von Workflowschema- und instanz gebraucht. Die Workflows werden in GroupProcess in Dokumenten gespeichert. In so einem Dokument befinden sich also die Workflowinstanz und das

Modell zuzüglich Zustandsinformationen. Zusätzlich wird auch das Modellierungswerkzeug selbst in das Dokument als serialisiertes Java-Objekt integriert. Dadurch kann das Dokument über verschiedene Kanäle von Benutzer zu Benutzer gelangen. Dieser braucht nur die Java-Plattform auf seinem lokalen Rechner und kann den Workflow dank des mitgelieferten Werkzeugs ändern oder erweitern.

6 Freiheitsgrade und Strukturänderungen durch Meta-Modell

In Joeris (2000) wird ein umfassender Ansatz für flexibles und adaptives Workflowmanagement vorgestellt. Die Repräsentation von Workflows wird auf Basis eines objektorientierten Meta-Modells vorgenommen. Die Ausführung von Workflows basiert auf dem Versenden von Ereignissen an Aufgaben bzw. Aktivitäten und das Reagieren auf Ereignisse anhand von Regeln.

In diesem Ansatz wird folgende Einteilung der Flexibilität vorgenommen.

A Priori Flexibilität: Die A Priori Flexibilität setzt in Fällen ein, wo die Struktur eines Workflows vorab partiell determiniert werden kann. Dies umfasst eine Erweiterung der Modellierungssprache für Ausnahmebehandlungen und der expliziten Modellierung von Freiheitsgraden für die Ausführung.

A Posteriori Flexibilität: Die A Posteriori Flexibilität kommt zum Einsatz, wenn die Struktur des Workflows überhaupt nicht vorab determiniert werden kann. In solch einem Fall muss dies zur Laufzeit geschehen. Sie umfasst neben Strukturänderungen- und erweiterungen auch das Tolerieren von Abweichungen zwischen Workflowausführung und Workflowmodell.

Abbildung 4 zeigt ein Beispiel für die A Priori Flexibilität. Hier sollen zwei Freiheitsgrade in einen einfachen Workflow modelliert werden. Die beiden Aufgaben Funktionstest und Ergonomieprüfung sollen auf keinen Fall parallel ausgeführt, eine Ausführungsreihenfolge allerdings offengelassen werden. Dieser Freiheitsgrad wird hier mit Exclusion bezeichnet. Der zweite Freiheitsgrad heißt SoftSync. Er soll ermöglichen, dass die Abnahme beginnen kann, auch wenn die Ergonomieprüfung nicht ausgeführt worden ist. Diese beiden Freiheitsgrade wurden in Form von neuen Kontrollstrukturen für Workflowmodellierer angegeben. Dabei wurden Abhängigkeiten zwischen den Zustandsautomaten der einzelnen Aufgaben in Form

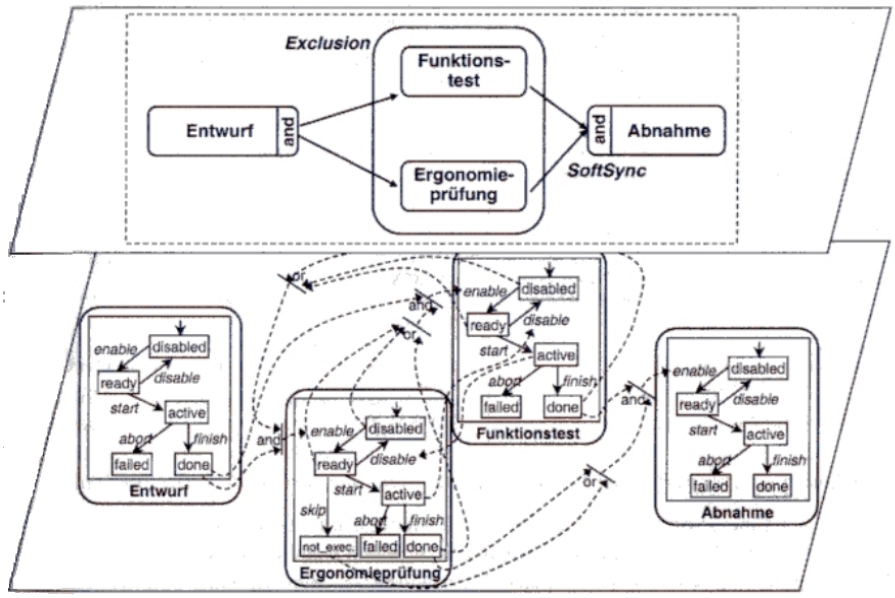

Abbildung 4: Beispiel für A Priori Flexibilität

von Regeln (hier nicht dargestellt) definiert. Diese Eigenschaft des Meta-Modells erlaubt es feingranulare Abhängigkeiten zwischen Aufgaben zu definieren.

Die A Posteriori Flexibilität wird auf zwei Weisen unterstützt. Zum einen werden Mechanismen bereitgestellt, mit denen ein Workflowmodellierer Teile einer Workflowinstanz suspendieren und eine Neumodellierung vornehmen kann. Zum anderen werden Operationen auf Basis von Graphtransformationen bereitgestellt, mit denen Workflowschemaänderungen automatisch durchgeführt werden können. Wird ein Workflowschema zur Laufzeit verändert, so hat dies Auswirkungen auf die einzelnen Instanzen. Damit diese die Änderungen reflektieren können werden entsprechende Migrationsstrategien- und transformationen bereitgestellt.

7 Master Thesis

Die Ansätze 3,4 und 5 ermöglichen es die Struktur des Workflows zur Laufzeit zu ändern. Dies ist eine wichtige Voraussetzung für die Unterstützung von Ad-Hoc Workflows. Allerdings basiert die Auslösung einer Änderung immer noch auf dem Menschen selbst. Dies spricht eigentlich gegen die Philosophie eines Workflowmanagementsystems, dessen Aufgabe es ist die Arbeitsabläufe zu steuern. Initiiert der Mensch die Änderungen an dem Workflow so

wird dem System diese Entscheidung abgenommen und das Ergebnis ist in der Regel nicht mehr als eine strukturierte Dokumentation des Workflows.

Ein Ansatz diesen Nachteil aufzuheben ist die Erstellung eines Ad-Hoc Workflows als Konfigurationsproblem zu betrachten. Ein Konfigurationsproblem ist die Auswahl und Zusammenstellung von Komponenten aus einer Menge, durch die eine Benutzerspezifikation erfüllt wird (vgl. Stefik (1995)). Konfigurationsaufgaben reichen von Auswahl von Elektronikbauteilen für den Zusammenbau eines Rechners bis hin zur Erstellung einer Behandlung eines Patienten. Der Author ist der Meinung, dass auch die Erstellung eines Workflows als Konfigurationsaufgabe gesehen werden kann.

Dazu müßte Anwendungswissen über den Ablauf verschiedener Geschäftspraktiken- und abläufe über eine Wissensbasis zugänglich gemacht werden. Die Konfiguration eines Workflows kann dann über Ableitungen von dieser Wissensbasis durch eine Inferenzmaschine erreicht werden. Allerdings muss beachtet werden, dass nicht nur die Menge an erforderlichen Komponenten (hier Aufgaben), sondern auch ihre Verknüpfungen durch Kontrollstrukturen herausgefunden werden muss.

Ist schließlich ein Workflow konfiguriert und gestartet kann beim Übergang von einer Aufgabe zur nächsten die Ziele des Workflows auf Konsistenz mit dem Anwendungswissen überprüft werden. Hat sich die Geschäftswelt geändert, so sollte dies im Anwendungswissen reflektiert werden. Wird eine Inkonsistenz festgestellt, so wird der Rest des Workflows erneut konfiguriert. Dies fällt dann in den Problembereich der Rekonfiguration.

8 Zusammenfassung

In diesem Bericht wurden fünf Ansätze zur Realisierung von flexiblen und Ad-Hoc Workflows kurz vorgestellt. Die Ansichten reichten dabei von Ad-Hoc Routing und Ressourcenzuteilung über eine verteilte Ausführung von Workflows über Agenten, bis hin zu der Möglichkeit die Struktur des Workflows komplett zur Laufzeit entstehen zu lassen. Bei diesen Ansätzen wurde ein Defizit in Bezug auf die Steuerung durch das Workflowmanagementsystem festgestellt. Die Initiative einen Workflow ad-hoc zusammenzustellen oder zu verändern ging immer vom Menschen selbst aus. Um diesen Mangel zu beheben wurde die Idee für eine Master-Thesis kurz vorgestellt.

Literatur

[Eric Y.Shan und Shan 2004] ERIC Y.SHAN, Fabio C. ; SHAN, Ming-Chien: Adaptive Process Management. In: *Workflow Handbook 2004*, 2004

[Huth und Nastansky] HUTH, Carsten ; NASTANSKY, Ludwig: Group Process: Partizipatives, verteiltes Design und simultane Ausführung von Ad hoc Geschäftsprozessen. – Forschungsbericht

[Jie Meng] JIE MENG, Stanley S.: An Ad-Hoc Workflow System Architecture Based on Mobile Agents and Rule-Based Processing. – Forschungsbericht

[Joeris 2000] JOERIS, Gregor: *Flexibles und adaptives Workflowmanagement für verteilte und dynamische Prozesse*. 2000

[Stefik 1995] STEFIK, Mark: *Introduction to Knowledge Systems*. 1995

[Voorhoeve und van der Aalst 1997] VOORHOEVE, M. ; AALST, W. van der: Ad-hoc Workflow: Problems and Solutions. 1997. – Forschungsbericht