

Universität Paderborn

Fakultät für Elektrotechnik, Mathematik und Informatik

Entwurf einer Workflowkomponente für das Dokumenten- und Wissensmanagementsystem VKC

Studienarbeit

vorgelegt von
Andreas Koop

Eidesstattliche Erklärung

Ich erkläre hiermit eidesstattlich, dass ich die vorliegende Arbeit selbständig und ohne fremde Hilfe angefertigt und alle Abschnitte, die wörtlich oder annähernd wörtlich aus einer Veröffentlichung entnommen sind, als solche kenntlich gemacht habe, ferner, dass die Arbeit noch nicht veröffentlicht und auch keiner anderen Prüfungsbehörde vorgelegt worden ist.

Andreas Koop

Eingereicht am:

30. September 2004

von:

Andreas Koop

Bussardweg 26

33813 Oerlinghausen

Matrikelnr: 6090183

E-Mail: kopyy@uni-paderborn.de

bei:

Dr. rer. nat. Bernd Kleinjohann

Prof. Dr. Wilhelm Schäfer

betreut von:

Dipl.-Inf. Dietmar Nolte

Dipl.-Inf. Lars Werner

Inhaltsverzeichnis

1	Einleitung	6
2	Zielsetzung	8
3	Existierende Lösungsansätze	11
3.1	Hartcodierte Lösung	11
3.2	Regelbasierte Programmierung	11
3.3	Workflow-Technologie	13
4	Workflow Management	15
4.1	Entwicklung und Abgrenzung	15
4.2	Aufgaben und Ziele	18
4.3	Definitionen	19
4.3.1	Grundlegende Begriffe	20
4.3.2	Struktur von Prozessen	23
4.3.3	Workflowtypen	24
4.3.4	Klassifikation von Workflow Systemen	26
4.4	Standardisierung: Die Workflow Management Coalition	27
4.4.1	Die Workflow Management Coalition	27
4.4.2	Funktionen eines Workflow Management Systems	28
4.4.3	Generische Struktur von Workflow Management Systemen	29
4.4.4	Das Referenzmodell der WfMC	32
5	Technologieanalyse	36
5.1	Prozessbeschreibungs- und Ausführungssprachen	36
5.1.1	XML Process Definition Language (XPDL)	36
5.1.2	WSFL, XLANG, BPEL4WS	40
5.1.3	WSCI	45
5.1.4	Business Process Modelling Language (BPML)	46

5.1.5	Business Process Modelling Notation (BPMN)	48
5.2	Workflow Engines	49
5.2.1	Java Business Process Management (jBpm)	49
5.2.2	Enhydra Shark	51
5.2.3	Bonita	51
5.2.4	Sonstige	52
6	Evaluierung	53
6.1	Virtual Knowledge Center (VKC)	53
6.2	Bewertung von Workflow Beschreibungssprachen	54
6.2.1	XPDL	54
6.2.2	WSFL, XLANG, BPEL4WS, WSCI	55
6.2.3	BPML	56
6.2.4	BPMN	56
6.3	Fazit	57
7	Realisierung	58
7.1	Konzeption und Design	58
7.1.1	Architektur	58
7.1.2	Anwendungsfälle	60
7.1.3	Systemanforderungen	61
7.2	Implementierung	62
7.2.1	Entwicklungsumgebung und Werkzeuge	62
7.2.2	Umsetzung	63
7.2.3	Integration in VKC	64
7.3	Praxisbeispiel	65
7.4	Test	66
7.5	Ergebnis	68
8	Zusammenfassung und Ausblick	73
Anhang A	Abkürzungen	76
Anhang B	Erstellung einer XPDL-Prozessdefinition mit JaWE	77
B.1	Paket-Attribute	77
B.1.1	Publication Status	77
B.1.2	Conformance Class	78
B.2	Zuordnung von Workflow Teilnehmern	78
B.3	Aktualisierung und Ansicht von Variablen	79

Anhang C Konfiguration	80
C.1 VKC	80
C.2 Shark Workflow Engine	80
Anhang D Inhalt der beiliegenden CD	82
Literaturverzeichnis	83

1 Einleitung

Noch immer steigt die Menge der zu bewältigenden Papierflut an. Weltweit werden täglich 200 Millionen Seiten Papier in Ordnern abgeheftet und mehr als 250 km neue Aktenordner angelegt. Diese Menge an papierbasierter Information wird noch durch die stark wachsende Zahl elektronischer Dokumente ergänzt, so dass sich die zu verarbeitende Informationsmenge alle 2-6 Jahre verdoppelt. Durch diese gigantische Flut von Informationen wird die bedarfsgerechte Verteilung und Koordination erheblich erschwert.

Zudem sind Informationen Kapital. Das heißt, je mehr Menschen damit arbeiten, es nutzen und gezielt einsetzen, desto wertvoller wird es für den, der es besitzt. Bleibt die Informationsmenge jedoch in Ruhe, so wird sie zu totem Kapital. Dabei liegt es nicht an fehlendem Willen, weswegen Informationen ungenutzt bleiben, sondern meist an der wenig ausgereiften Technik, welche den Datenfluss behindert.

Zwar haben Dokumenten- (DMS) und Wissensmanagementsysteme (WMS) in den letzten Jahren enorme Abhilfe geleistet, doch fehlt es diesen Systemen in der Regel an einer Koordination und Automation des Informationsflusses. Einfache Arbeitsabläufe wie z. B. das Korrekturlesen und damit verbundene Freigeben bzw. Abweisen eines Dokumentes werden kaum unterstützt und müssen folglich manuell durchgeführt werden. Typischerweise erfordert so ein Arbeitsschritt die Bearbeitung einer Mappe mit diversen Formularen und Dokumenten. Hat der Beteiligte die Dokumente durchgesehen und die sachbezogenen Stellen im Formular ausgefüllt, wird die Mappe an die Person weitergereicht, welche die nächste Aufgabe im jeweiligen Arbeitsablauf vornimmt. Ein derartiges Vorgehen setzt eine intensive Schulung der Beteiligten voraus, oder die Mappe enthält Instruktionen in Form einer angehängten Notiz über den weiteren Verlauf. Um den Fortschritt eines komplexeren Arbeitsauftrags zu verfolgen, muss jemand durch die Abteilung gehen und sich nach der aktuellen Lage erkundigen. Es gibt keine einfache Möglichkeit, festzustellen, ob Aufgaben zu spät erledigt werden oder gar unter

den Tisch fallen.

Um diesem Dilemma zu entgehen, wird *Workflow Management* als Schlüsseltechnologie gesehen [KLR95, KR98] und bereits seit längerem eingesetzt. Jedoch bieten Workflow Management Systeme wenig bis keine Dokumentenmanagement Funktionalität, obwohl eine natürliche Synergie zwischen beiden Technologien besteht [Kou95]. Ein System, das beide Verfahren miteinander vereint, ist erstrebenswert.

2 Zielsetzung

Angesichts der bestehenden Problematik soll das Dokumenten- und Wissensmanagementsystem VKC¹ um eine Workflowkomponente erweitert werden, welche sowohl die Ausführung rudimentärer Geschäftsprozesse unterstützt als auch die Verarbeitung von Dokumenten unter mehreren Benutzern in einer Netzwerkumgebung steuert.

Die folgenden Szenarien sollten nach Möglichkeit unterstützt werden:

Dokumenten-Freigabe : Der *Autor* erstellt ein Dokument, der *Redakteur* erhält eine Nachricht über das neue (oder auch geänderte Dokument) und kann die Freigabe, d. h. Live-Stellung per Mausklick durchführen oder gegebenenfalls das Dokument zur Korrektur an den *Autor* zurückweisen.

Projektarbeit : Der *Projektleiter* erstellt eine Spezifikation und weist diese einem *Programmierer* zu. Nachdem dieser die Aufgabe implementiert hat, wird ein *Tester* benachrichtigt, um die Implementierung zu testen. Falls der Test fehl schlägt, wird die Aufgabe erneut an den *Programmierer* zur Fehlerbehebung weitergeleitet. Die Interaktion zwischen dem *Programmierer* und *Tester* dauert so lange an, bis der Test erfolgreich abgeschlossen wird. Anschließend wird der *Projektleiter* über das Ergebnis informiert und kann die Spezifikation als implementiert abhaken.

Dynamisches Besprechungsprotokoll : Während bei einem gewöhnlichen Protokoll nicht selten wichtige Aufgaben in Vergessenheit geraten, können bei einem dynamischen Besprechungsprotokoll die erörterten und anstehenden Aufgaben unmittelbar an die teilnehmenden *Personen* zugewiesen werden. Muss die Aufgabe zu einem bestimmten Termin fertiggestellt sein, so kann sie darüberhinaus mit einer Deadline verknüpft und somit überwacht werden. Sollte der Termin nicht eingehalten werden können, wird der *Projekt-*

¹Virtual Knowledge Center, <http://www.vkc.info>

leiter über die Eskalation der Deadline umgehend verständigt und kann die damit verbundenen Maßnahmen einleiten.

Nach der Besprechung hat jeder Benutzer seine persönliche Arbeitsliste und kann sich entsprechend der Deadline oder Priorität einer konkreten Aufgabe an die Arbeit machen.

Auf diese Weise gehen keine wichtigen Aufgaben mehr verloren.

Lesebestätigung : Ein *Autor* erstellt ein Dokument und möchte, dass es von einem *anderen Benutzer* oder einer *Gruppe* gelesen wird. Dies ist sinnvoll bei gemeinschaftlichen Arbeiten an einem Projekt bzw. in einer Arbeitsgruppe.

Elektronische Laufmappen : Ähnlich einem gewöhnlichen Ordner enthält eine Laufmappe eine Menge von Dokumenten und Notizen. Darüberhinaus ist der Ordner jedoch mit einer Aufgabenliste verknüpft, bestehend aus einem Benutzer und einer Aufgabenbeschreibung, z. B. Review eines Dokumentes oder Ausfüllen eines Formulars. Nachdem der Initiator die Aufgabenliste erzeugt hat, wird die Laufmappe an den nächsten Bearbeiter in der Liste weitergeleitet.

Zu jedem Zeitpunkt ist der Ordner jeweils nur für den aktuellen Bearbeiter und den Initiator sichtbar, welcher den aktuellen Status verfolgen und gegebenenfalls die Aufgabenliste verändern kann. Sobald die letzte Aufgabe bearbeitet ist, wird der Umlauf Ordner an den Initiator weitergereicht und der Gesamtprozess gilt damit abgeschlossen.

Darüberhinaus ist die Automatisierung rudimentärer Geschäftsprozesse vorstellbar, wie:

- Redaktions- oder Wissensakquisitionsprozesse
- Reise- und Urlaubsanträge
- Krankmeldungen
- Fehlerverfolgung und -aufklärung (Bugtracking)
- Bestellverfahren und Ausführung

Dies sind nur einige wenige Anwendungsbeispiele gängiger Geschäftsprozesse, die von einer Workflowkomponente gefördert werden sollten. Gute Workflow Managementsysteme müssen enorm *flexibel* sein, um sich veränderten Bedingungen und Marktsituationen in der Geschäftswelt anpassen zu können [BP98].

Neue Technologien, neue Gesetze und Marktanforderungen führen unmittelbar zu (strukturellen) Veränderungen einer Geschäftsprozessdefinition² [vdABV⁺99].

Aufgabenstellung

Einen wesentlichen Teil der Arbeit beansprucht die Analyse bestehender Standards im Bereich der *Prozessmodellierung und -beschreibung*. Welche Sprache eignet sich am besten für den Einsatz im Umfeld des Dokumentenmanagements? Welche Unterstützung gibt es bei der Modellierung eines Prozesses in einer bestimmten Sprache?

Der Schwerpunkt liegt auf der Konzeption und Implementierung einer Workflowkomponente, die als optionale Erweiterung an das bestehende System angebunden werden kann. Dabei soll kein graphischer Editor zur Modellierung eines Workflows erstellt, sondern eine Schnittstelle geschaffen werden, die es ermöglicht, eine Workflowbeschreibung zu importieren und dynamisch an bestehende Kategorien, Benutzer und Dokumente anzubinden.

Gliederung

Für eine erfolgreiche Implementierung einer Workflowkomponente gilt es die existierenden Standards zu analysieren, um sich für ein Beschreibungsformat festzulegen. Die vorliegende Studienarbeit gliedert sich aus diesem Grund in 2 grobe Teile: einen Theoretischen und einen Praktischen. Im theoretischen Teil werden zunächst existierende Lösungsansätze diskutiert (Kapitel 3), grundlegende Begriffe zum Thema Workflow Management erläutert (Kapitel 4) und es werden im besonderen Maße existierende Sprachen zur Beschreibung von Prozessen analysiert und beurteilt (Kapitel 5). Außerdem werden einige Open Source Workflow Engines untersucht und ihre grundlegende Architektur vorgestellt (Abschnitt 5.2). Die Evaluierung der gewonnenen Erkenntnisse wird in Kapitel 6 vorgenommen.

Im praktischen Teil (Kapitel 7) wird zum einen die grundlegende Konzeption und der Entwurf der entwickelten Workflowkomponente illustriert (Abschnitt 7.1), zum anderen die Implementierung (Abschnitt 7.2) beschrieben und das Ergebnis diskutiert. Den Abschluss bildet eine Zusammenfassung der Erkenntnisse aus der eigenen Arbeit und ein Ausblick für weitere Forschungsarbeit (Kapitel 8).

²Beschreibung eines Geschäftsprozesses in textueller/graphischer Form

3 Existierende Lösungsansätze

Für die Implementierung einer Workflowkomponente existieren bereits mehrere Ansätze. In diesem Kapitel werden sie genauer vorgestellt und im Hinblick auf die Anforderungen beim Einsatz in einem Dokumentenmanagementsystem erörtert.

3.1 Hartcodierte Lösung

Wie der Name erahnen lässt, handelt es sich bei diesem Ansatz darum, den Ablauf eines Geschäftsprozesses hart zu codieren. Geschäftsprozesse und Abläufe werden in der Anforderungsanalyse in Form von Prozessdiagrammen beschrieben und anschließend in Programmcode umgesetzt [RT04]. Einfache (wenige) Prozesse können schnell und ohne großen Aufwand implementiert und deren Ausführung überwacht werden.

Kritik

Die Änderung oder das Hinzufügen eines neuen Workflows wird zu einer haarsträubenden Prozedur: Der Programmcode muss geändert, neu geschrieben oder angepasst und anschließend kompiliert werden. Ein derartiges Verfahren ist nicht nur fehleranfällig, sondern bedarf der ständigen Wartung durch qualifizierte Mitarbeiter. Für heutige Unternehmen ist das eine unzumutbare und vor allen Dingen kostspielige Angelegenheit, für einen produktiven Einsatz gar undenkbar.

3.2 Regelbasierte Programmierung

Es ist allgemein bekannt, dass monolithische Systeme, in denen jegliche Geschäftslogik im Programmcode fest verankert ist, früher oder später die Weiterentwicklung erschweren und zu einer Langzeit-Wartung führen [Bro94]. Aus diesem Grund sind bereits in den 80ern *Regelbasierte Programmiersprachen* entworfen

worden, um den Programmierer von irrelevanten Implementierungsdetails zu befreien und gleichzeitig eine weitere Abstraktionsebene zur Objektorientierten Programmierung zu schaffen [Tec96]. Eine der ersten Regelbasierten Sprachen war OPS5 [CW88]. In einer solchen Sprache fungiert eine *Regel* als Anweisung, bestehend aus einer *Bedingung* und einer *Aktion*, welche spezifiziert, was zu tun ist, sobald die Bedingung von bestimmten Daten erfüllt wird. Derartige Regeln können z. B. mit einem graphischen Editor erstellt und anschließend mit der eigentlichen Anwendung verknüpft werden. Dadurch ist es möglich, den Programmablauf zur Laufzeit deklarativ zu verändern.

Heutzutage spricht man in diesem Zusammenhang auch von der *Business Rule-Technologie*. Das Grundkonzept bleibt das gleiche: IF-THEN-Statements aus dem Applikations-Code herauszuziehen und extern zu pflegen. In [uFS04] findet sich dazu ein verdeutlichendes Beispiel:

Viele solcher Statements im Code einer Applikation beziehen sich auf die Geschäftslogik eines Unternehmens. Im Telekommunikationsbereich könnte das beispielsweise eine Regel über Rabatte in Verbindung mit einer bestimmten Gesprächsdauer sein: „Wenn die Dauer eines Telefongesprächs größer ist als 5 Minuten, dann gewähre 5 Prozent Rabatt auf die Minutengebühr“.

Die extern gehaltenen Regeln werden zur Laufzeit von einer so genannten Regelmaschine (Rule-Engine) geladen und ausgeführt. Die Regelmaschine selbst ist ihrerseits in der eigentlichen Applikation eingebettet.

Eines der führenden Produkte in diesem Bereich ist *ILOG JRules*¹. Es bietet ein umfangreiches Paket mit vielen nützlichen Tools zur Entwicklung einer Regelbasierten Applikation. Unter anderem ist ein graphischer Editor verfügbar, mit dem Regeln zu einem *Ruleflow* verknüpft werden können. Abbildung 3.1 zeigt den Ruleflow Editor von ILOG JRules. Für die kommenden Jahre sehen die Analysten der Gartner Group Business Rule Software auf dem Weg zur allgemeinen Verbreitung in Unternehmen. Nach Ansicht von [Gar03] wird dieser Softwarebereich darüberhinaus beachtliche Auswirkungen auf Workflow Management haben.

Kritik

Die erhoffte Verbreitung ist bislang (zumindest) in Deutschland ausgeblieben. Grund dafür könnte die mangelnde Interoperabilität zwischen den verschiedenen

¹<http://www.ilog.de/products/jrules/>

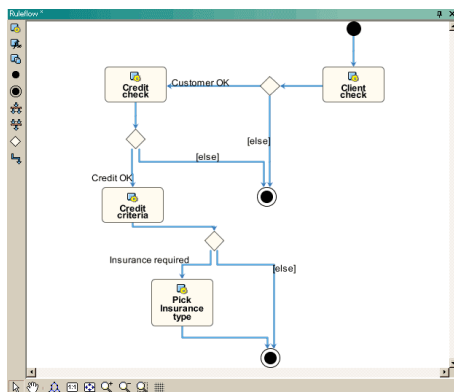


Abbildung 3.1: Ruleflow Editor von ILOG JRules (Quelle: [JRu])

Business Rule Systemen sein. Jeder Anbieter implementiert eine eigene Beschreibungssprache für die Regeln und Ruleflows. Zwar gibt es meistens umfangreiche APIs für die Integration in eine bestehende Applikation, es existieren jedoch keine standardisierten Schnittstellen oder Austauschformate. Ein Austausch der Rule-Engine ist nicht ohne größeren Aufwand zu bewältigen.

Zudem sind die bestehenden Werkzeuge oft sehr komplex gehalten und bedürfen einer längeren Einarbeitungsphase. Benutzer eines Dokumentenmanagementsystems, die „mal eben“ einen Workflow auf einem Dokument definieren wollen, sind schnell überfordert.

Außerdem können mit der Rule-Technologie keine hierarchischen Strukturen modelliert werden.

3.3 Workflow-Technologie

Durch die Einführung von Workflow-Technologie lassen sich die in den vorhergehenden Abschnitten 3.1 und 3.2 beschriebenen Probleme und Schwierigkeiten vermeiden: Die graphische Modellierung von Prozessen und die einfache, auch für Nicht-Entwickler verständliche Darstellung, erlauben es, Abläufe ad-hoc zu modellieren und in ausführbare Prozessdefinitionen umzusetzen. Die so entstan-

denen Prozessdefinitionen können jederzeit in ihrer graphischen oder textuellen Form eingesehen, überprüft und gegebenenfalls modifiziert werden. Abbildung 3.2 veranschaulicht das Vorgehen beim Modellieren und Ausführen von Prozessen.

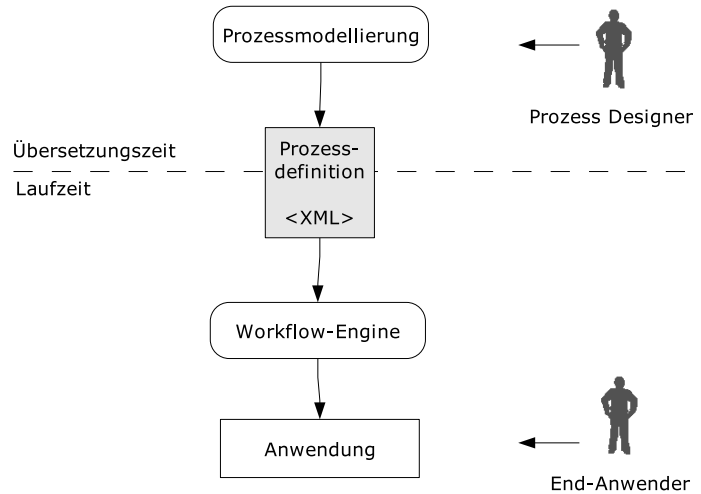


Abbildung 3.2: Prinzipielle Funktionsweise eines prozessbasierten Ansatzes

Der *Prozessdesigner* modelliert mit Hilfe eines graphischen Werkzeugs den gewünschten Prozess und erzeugt als Ergebnis die daraus resultierende Prozessdefinition. Diese wird anschließend in einer umgebenden Applikation „deployed“ und vom Laufzeitsystem interpretiert, um ausführbare Instanzen von Prozessen zu generieren. Die Workflow-Engine übernimmt dabei die Erzeugung und Ausführung von Prozessinstanzen. Auf diese Weise können Ablauf- und Anwendungslogik getrennt und folglich Anwendungssysteme realisiert werden, die sich (zumindest vom Prinzip her) sehr viel einfacher an geänderte Geschäftsprozesse anpassen lassen [JBH97].

Kritik

Trotz der Workflow Management Coalition (WfMC²), die sich seit 1993 für die Etablierung von Standards im Bereich der Workflow-Technologie einsetzt, sind mehrere Beschreibungssprachen für Prozessdefinitionen hervorgegangen. Diese Tatsache erschwert immer noch eine reibungslose Interoperabilität und Konnektivität zwischen unterschiedlichen Workflow Management Systemen. Unternehmen, die ihre Geschäftsprozesse mit einer Workflowkomponente unterstützen wollen, haben die Qual der Wahl. Eine kostenintensive Analyse der gegebenen Anforderungen ist unvermeidbar, um sich für das richtige System zu entscheiden.

²<http://www.wfmc.org/>

4 Workflow Management

In diesem Kapitel soll eine Einführung in das Themengebiet von *Workflow Management* gegeben werden. Dazu wird im ersten Abschnitt die Geschichte und Entwicklung von Workflow Management vorgestellt. Abschnitt 4.2 befasst sich anschließend mit den Aufgaben und Zielen, die mit Workflow Management einhergehen. Der letzte Abschnitt führt grundlegende Definitionen für das Verständnis von Workflow Management ein.

4.1 Entwicklung und Abgrenzung

Es gibt wohl 2 große Entwicklungen in der Informationstechnologie, die als Auslöser für die Einführung von Workflow Management in Betracht kommen. Die erste Entwicklung ist der enorme Fortschritt von Hard- und Software Technologien in den letzten 2 Jahrzehnten, welcher zu einer Umwälzung in der Büroarbeit führte. Immer leistungsfähigere PCs und Workstations konnten zu niedrigen Kosten hergestellt und in der Büroarbeit eingesetzt werden. Darüberhinaus ermöglichten neue Netzwerktechnologien und Datenbanksysteme die Entwicklung von integrierten Softwaresystemen. Als Folge dieser Errungenschaften wurden anspruchsvollere Anwendungen wie Textverarbeitung, Tabellenkalkulation und diverse Bildverarbeitungsprogramme entwickelt. Die Gesamtproduktivität im Anwendungsbereich ist – proportional zu diesen Verbesserungen gesehen – auf Grund mangelnder Integration der neuen Softwareprodukte dennoch nicht gestiegen. Um bessere Ergebnisse zu erzielen, war eine Änderung in der Entwicklung von Anwendungssystemen notwendig.

Diese Notwendigkeit wurde zum zweiten Auslöser für den Einsatz von Workflow Management. Das bis dahin bestehende Ziel, Teile von Anwendungssystemen zu automatisieren, wurde durch den neuen Ansatz einer umfassenden Entwicklung von integrierten Anwendungssystemen ersetzt. Dieses neue Ziel brachte eine Verlagerung von der aufgaben- bzw. datenorientierten Sicht zur prozess- bzw. ar-

beitsorientierten Sicht in der Entwicklung von Softwaresystemen. Dabei steht bei der aufgaben- bzw. datenorientierten Sicht die Integration der Daten im Vordergrund, wohingegen sich die prozess- bzw. arbeitsorientierte Sicht auf die Arbeit als Ganzes konzentriert. Mit diesem Ansatz wird die Integration von Programmen, Prozessen, Daten, Dokumenten, Personen, Organisationseinheiten etc. erstrebt, was zur Optimierung von Prozessen führt – Minimierung der Kosten bei gleichzeitiger Effizienzsteigerung [JB96].

Die beschriebenen Veränderungen in der Informationstechnologie führten unmittelbar zur Entwicklung neuer Softwaresysteme, welche einige Aspekte des Workflow Managements aufweisen. Einige dieser neuen Technologien sollen nun etwas genauer vorgestellt und voneinander abgegrenzt werden.

Office Automation

Workflow Management kann als Folge der *Office Automation* angesehen werden, welche Anfang der 1970er ihren Lauf nahm. Beide Technologien beabsichtigen die Automatisierung von Arbeitsvorgängen. Der Unterschied besteht darin, dass Office Automation auf die Automatisierung von bestimmten Aufgaben abzielt, wohingegen Workflow Management die Automatisierung der Kontrolle von Aufgaben während eines Geschäftsprozesses beabsichtigt. Trotz der unterschiedlichen Konzepte gibt es eine Reihe von Ideen wie z. B. das Scheduling von Aktivitäten und Management von Aufgaben, die sich in Workflow Management Systemen wiederfinden [BP84].

Dokumenten Management

Die weite Verbreitung der Computer Technologie in der Büroarbeit führte zur Ersetzung von papierbasierten Dokumenten durch elektronische Dokumenten. Diese mussten effizient verwaltet werden und so entstand die erste Generation von *Dokumenten Management Systemen*. Passive¹ Systeme, welche die Anfrage, Sperre oder Freigabe eines bestimmten Dokumentes unterstützten.

Die Weiterentwicklung führte zu den sogenannten *Aktiven Dokumenten Management Systemen*, welche mit zeitabhängigen Randbedingungen glänzten. Abhängig von der aktuellen Zeit konnte ein Dokument z. B. eingeblendet oder nach einem bestimmten Zeitintervall gelöscht werden. Im vereinfachten Sinne war das ein erster Schritt in Richtung Workflow Management. Aktive Dokumenten Management

¹Reaktion auf direkte Benutzereingaben

Systemen werden daher auch als Vorfahren von dokumentenorientierten Workflow Management Systemen gesehen [JB96].

E-Mail

Moderne E-Mail Systeme werden benutzt, um Informationen entsprechend der Eigenschaften einer Person (z. B. Rolle, Arbeitsgruppe) innerhalb einer Organisation zu verteilen. Dies ist zugleich auch ein wesentliches Charakteristikum von Workflow Management.

Groupware

Mit *Groupware* oder auch *Kollaborationssoftware* bezeichnet man rechnergestützte Systeme, welche eine Gruppe von Leuten bei einer gemeinsamen Aufgabe unterstützen und Zugang zu einer gemeinsamen Arbeitsumgebung verschaffen [EGR91]. Im engeren Sinne sind Groupware Systeme verteilte Anwendungen, die gleichzeitig einen Zugriff auf Daten und Dokumente ermöglichen. Diese können von Benutzern, die sich an unterschiedlichen Orten befinden, gemeinsam bearbeitet werden. Nicht zu verwechseln ist dabei CSCW²: Eine Forschungsdisziplin, in der das Design, die Anpassung und Benutzung von Groupware untersucht wird.

Während Groupware primär die *Kooperation* über zeitliche und räumliche Distanzen hinweg unterstützt, dienen Workflow Management Systeme zur *Koordination* meist strukturierter, wiederkehrender Prozesse [wik04]. Dennoch wird Groupware als ein geeignetes, übergeordnetes Anwendungsgebiet für Workflow Management angesehen [Boc93].

Software Entwicklungsmanagement

Software Entwicklungsmanagement koordiniert die Entwicklung von Software. Es umfasst hauptsächlich folgende 3 Phasen: Modellierung, Analyse und Ausführung des Modells [JB96]. Nicht selten werden in einem Entwicklungsprozess Workflow Funktionalitäten bereitgestellt, um verschiedene Entwicklungsaufgaben und Informationen innerhalb eines Teams zu verteilen.

Business Process Reengineering (BPR)

Mit ändernden Bedingungen und Marktsituationen in der Geschäftswelt sind viele Unternehmen gezwungen, ihre Geschäftsprozesse neu zu strukturieren. *Business*

²Computer Supported Cooperative Work

Process Reengineering (BPR) stellt Werkzeuge zur Verfügung, welche die Analyse, Modellierung und Definition von Geschäftsprozessen unterstützen. Der Fokus dabei ist jedoch auf den Geschäftsprozess als Ganzes gerichtet und nicht auf die koordinierte Ausführung einzelner Aufgaben. Darüberhinaus berücksichtigt BPR verschiedene Sichten eines Geschäfts: Technologische, wirtschaftliche, informationsorientierte Sicht etc.

Die grundlegende Idee von BPR führte zur Entwicklung von Workflow Management Systemen [JB96].

Enterprise Application Integration

Während beim traditionellen Workflow Management das Dokument als essentielle Größe betrachtet wird, steht bei EAI das *Application Programming Interface (API)*, welches eine Anwendung für ihr Umfeld zugänglich machen kann, im Vordergrund. EAI definiert einen Workflow als festgelegte Sequenz von API-Aufrufen auf verschiedenen Anwendungen. Statt Workflow oder Geschäftsprozess wird daher der Begriff *Cross-Application Process* oder gar *Cross-Application Procedure* vorgeschlagen [AG03].

Business Process Management (BPM)

Ein Geschäftsprozess beschreibt beim Workflow Management den Fluss von Dokumenten, die über Abteilungen und Organisationen hinweg ausgetauscht werden und deren Aktivitäten bestimmten Personen oder Rollen einer Organisation zugeordnet werden. Für das Business Process Management (BPM) ist dieser Ansatz zu restriktiv, da Geschäftsprozesse auch ohne Dokumente funktionieren müssen. Nach dem Workflow Handbook 2004 wird BPM als Konvergenz von Workflow, EAI und dem Web verstanden[Fis04].

4.2 Aufgaben und Ziele

Mit dem Einsatz von Workflow Management werden nach [wik04] allgemein folgende Ziele verfolgt:

- Verbesserung der Qualität von Prozessen
- Vereinheitlichung der Qualität von Prozessen
- Schnelle und effiziente Bearbeitung von Kundenaufträgen

- Verkürzung der Transport- und Liegezeiten
- Reduzierung der Bearbeitungszeiten und damit der Kosten
- Erhöhung der Verfügbarkeit von Informationen
- Vermeidung von Medienbrüchen

Die zu bewältigenden Aufgaben dabei sind nach [BS95]

- die Organisationsmodellierung
- die Definition von Vorgangstypen
- die Ausführung und Überwachung der Vorgänge
- das Vorgangsmanagement³

Die Einführung eines Workflow Management Systems in einer Organisation oder einem Unternehmen stellt einen erheblichen Aufwand dar, der durch den Einsatz des Systems gerechtfertigt werden muss. Grundsätzlich ergeben sich nach [BS95] jedoch folgende Vorteile:

Erhöhte Produktivität: Durch die Vermeidung von Transport- und Liegezeiten wird Zeit eingespart und Parallelarbeit ermöglicht.

Nachweisbarkeit: Alle Arbeitsabläufe und Geschäftsprozesse werden vom Workflow Management System dokumentiert und protokolliert.

Qualitätssicherung: Das System überwacht alle Arbeitsschritte und stellt sicher, dass sie zu einem festgesetzten Zeitpunkt erledigt werden, oder meldet, dass die Erledigung noch aussteht.

Auskunfts-fähigkeit: Der aktuelle Bearbeitungsstand einer Tätigkeit kann jederzeit ermittelt werden.

4.3 Definitionen

Im Laufe der Entwicklung haben sich zahlreiche Definitionen im Bereich des Workflow Managements entstanden. In diesem Abschnitt werden zunächst *grundlegende Begriffe*, die für das Verständnis von Workflow Management notwendig sind, eingeführt und ihre Beziehung untereinander aufgezeigt. Anschließend werden gängige *Workflowtypen* vorgestellt und zum Schluss eine Klassifikation von Workflow Management Systemen vorgenommen.

³Effiziente Verkürzung von Geschäftsprozessen

4.3.1 Grundlegende Begriffe

Die grundlegenden Begriffe und ihre Beziehungen untereinander sind in Abbildung 4.1 gemäß [Coa99a] dargestellt.

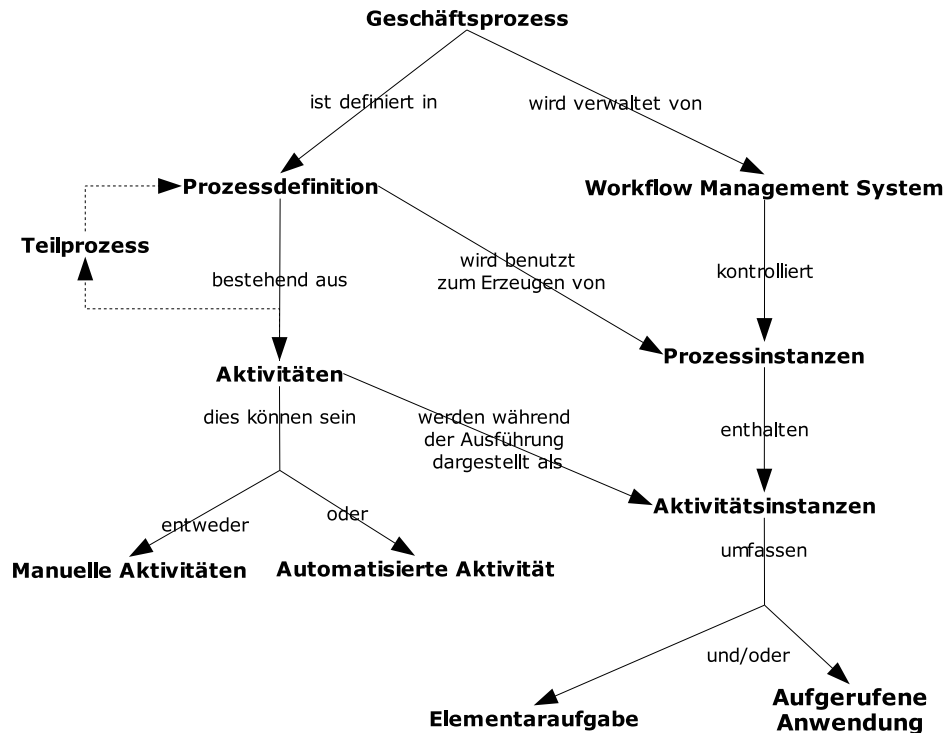


Abbildung 4.1: Beziehungen zwischen den grundlegenden Begriffen

Geschäftsprozess

Ein Geschäftsprozess ist eine Menge von fachlich zusammenhängenden *Aktivitäten*, die notwendig sind, um einen Geschäftsfall zu bearbeiten. Die einzelnen *Aktivitäten* können organisatorisch verteilt sein, stehen aber gewöhnlich in zeitlichen und logischen Abhängigkeiten zueinander [BG00].

Workflow

Der Begriff *Workflow* wird von der WfMC⁴ (dazu mehr in Kapitel 4.4) definiert als

„...Automatisierung (komplett oder nur teilweise) von Geschäftsprozessen, in denen Dokumente, Informationen oder Aufgaben von einem

⁴Workflow Management Coalition

Beteiligten zu einem anderen entsprechend gegebener Ablaufregeln weitergereicht werden, um eine bestimmte Aktion durchzuführen.“ [Coa99a]

Ein Workflow stellt also einen zumindest teilweise automatisierten Ablauf von Aktivitäten dar, die Bestandteil von Geschäftsprozessen sind. Im Allgemeinen sind Workflows arbeitsteilige Prozesse, welche die anfallenden Tätigkeiten von Personen und Softwaresystemen koordinieren [wik04].

Workflow Management (System)

Das Forschungsfeld *Workflow Management* befasst sich mit der Automatisierung von Arbeitsvorgängen. Dabei geht es um Handlungen wie das Organisieren, Planen, Entscheiden, Kontrollieren, Steuern und Führen eines Workflows [JBH97]. Für die Ausführung dieser Aufgaben ist das *Workflow Management System (WfMS)* verantwortlich. Es delegiert ausstehende *Aktivitäten* an die beteiligten Organisationseinheiten, Rollen, Personen und Applikationen nach den Vorgaben der *Prozessdefinition*. Weiterhin besteht ein Workflow Management System aus einer Entwicklungsumgebung, die der Definition von Prozessen dient und einer Laufzeitumgebung, welche die Durchführung von *Prozessinstanzen* steuert, Ressourcen, Applikationen, Daten, Dokumente für die Ausführung zur Verfügung stellt und den Status einer *Prozessinstanzen* zur Laufzeit protokolliert (nach H. Krallmann).

Prozessdefinition

Eine *Prozessdefinition* ist eine formale Beschreibung eines Prozesses in einer Form, welche die Modellierung und Ausführung durch ein Workflow Management System unterstützt. Sie besteht aus einem Netzwerk von *Aktivitäten*, verknüpft mit Informationen bezüglich der Teilnehmer, relevanten Daten und *aufrufenden Applikationen* [Coa99a].

Aktivität

Unter einer *Aktivität* versteht man die Beschreibung eines Arbeitsschrittes innerhalb eines Prozesses [Coa99a].

Obwohl dieser Begriff aus dem Workflow Management kaum wegzudenken ist, gibt es Überlegungen ihn durch *Zustand* oder *Aktion* zu ersetzen. Nach [Bae04] vermischt der Ausdruck den Unterschied zwischen *Zustand* und *Aktion*. Ein *Zustand* in einem Prozess spezifiziert ein Abhängigkeitsverhältnis zu einem externen Teilnehmer. Das bedeutet, dass während der Ausführungszeit der Prozess

solange angehalten wird bis der externe Teilnehmer das Workflow Management System über die Beendigung des Zustands benachrichtigt, z. B. das Warten auf eine Genehmigung. Eine *Aktion* hingegen ist ein Stück Programmlogik, das vom Workflow Management System ausgeführt wird, sobald ein festgelegtes Ereignis eintritt, z. B. das Versenden einer E-Mail, falls ein bestimmter Zustand einem Workflow Teilnehmer zugewiesen wird [Bae04].

Manuelle Aktivität

Als *manueller Aktivität* bezeichnet man eine Aktivität innerhalb eines Geschäftsprozesses, die nicht für die Automation geeignet ist und somit außerhalb des Workflow Management Systems durchgeführt werden muss. Solche Aktivitäten können in der Prozessdefinition enthalten sein, um z. B. die Modellierung eines Prozesses zu unterstützen. Sie sind allerdings nicht Teil des daraus resultierenden Workflows [Coa99a].

Automatisierte Aktivität

Eine *automatisierten Aktivität* charakterisiert eine Aktivität, die für Computerautomation geeignet ist. Die Automation erfolgt mit Hilfe eines Workflow Management Systems, das die Aktivität während der Ausführung des Geschäftsprozesses verwaltet. Automatisierte Aktivitäten werden auch als *Workflow-Aktivitäten* bezeichnet und erfordern menschliche und/oder maschinelle Ressourcen [Coa99a].

Prozess- / Aktivitätsinstanz

Eine Instanz ist die Repräsentation einer einzelnen Prozessausführung (*Prozessinstanz*), oder Aktivität (*Aktivitätsinstanz*) eines Prozesses, und deren assoziierten Daten. Jede Instanz repräsentiert einen eigenen Thread des Prozesses oder der Aktivität. Dieser kann unabhängig kontrolliert werden, besitzt seinen eigenen internen Zustand und eine extern sichtbare Identität [Coa99a].

Workflow Teilnehmer

Ein *Workflow Teilnehmer* ist eine Ressource, welche die Arbeit, die durch eine Workflow Aktivitätsinstanz repräsentiert ist, ausführt. Diese Arbeit besteht im Allgemeinen aus einer oder mehreren *Elementaraufgaben*. Normalerweise wird der Begriff Workflow Teilnehmer mit Human Resources in Verbindung gebracht. Es sind hier aber auch durchaus Maschinen gemeint, wie z. B. ein intelligenter Agent [GRV99].

Ein Workflow Teilnehmer wird entweder direkt in der Prozessdefinition identifiziert, oder die Identifikation erfolgt innerhalb der Prozessdefinition durch eine Referenz auf eine Rolle. Diese Rolle kann dann bei der Instanziierung durch eine oder mehrere Ressourcen gefüllt werden.

Elementaraufgabe (Work Item)

Eine *Elementaraufgabe (Work Item)* ist die Repräsentation der Arbeit, die von einem Workflow Teilnehmer gemacht werden muss. Aus einer Aktivität gehen normalerweise ein oder mehrere Elementaraufgaben hervor, die zusammen die Arbeit bilden, die von einem Benutzer in dieser Aktivität erledigt werden muss [Coa99a].

Aufgerufene Anwendung

Eine *Aufgerufene Anwendung* ist eine Workflow Applikation, die vom Workflow Management System aufgerufen wird, um eine Aktivität (zum Teil) automatisch auszuführen oder einen Workflow Teilnehmer bei der Bearbeitung seiner Elementaraufgaben zu unterstützen [Coa99a].

4.3.2 Struktur von Prozessen

Gemäß dem vorherigen Abschnitt ist ein Prozess ein formal beschriebener Vorgang, der sich aus Aktivitäten zusammensetzt. Bei dieser Zusammensetzung können verschiedene Verzweigungspunkte auftreten. Wenn der Kontrollfluss auf mehrere Aktivitäten aufgeteilt wird, die alle *gleichzeitig* ausgeführt werden können, spricht man von einem *AND-Split*. Der Punkt, an dem diese Aufspaltung wieder zusammengeführt wird, heißt *AND-Join* und wird auch als *Synchronisationspunkt* bezeichnet [vdA03]. Bei einer Verzweigung, von der jeweils nur ein Pfad ausgeführt werden muss, handelt es sich um einen *OR-Split*. Die Zusammenführung heißt in diesem Fall *OR-Join*.

Die hier vorgestellten Verzweigungsmöglichkeiten sind nur ein kleiner Teil der ca. 20 existierenden *Workflow Patterns*. Zur weiteren Vertiefung sei auf [vdABtHK00] verwiesen.

Um die Strukturierung von Prozessen zu erweitern, wurde das Konzept der *Sub- bzw. Teilprozesse* eingeführt. Ein Teilprozess ist ein Prozess, der innerhalb eines anderen Prozesses ausgeführt wird. An der Spitze einer so gebildeten Prozesshierarchie steht der *Top-Level-Prozess*.

Zusätzlich zu den Verzweigungen können in einer Prozessdefinition auch *Iterationen* vorkommen. Dies ist der Fall, wenn eine Aktivität solange wiederholt werden muss bis eine zuvor festgelegte Abbruchbedingung erfüllt ist.

4.3.3 Workflowtypen

Viele Jahre lang haben Wirtschaftsanalytiker und Autoren Workflows zu kategorisieren versucht. Obwohl solche Kategorien in Ungunst gefallen sind, sind sie dennoch sehr aufschlussreich und helfen die Unterschiede zwischen verschiedenartigen Workflow Systemen zu verstehen. In der Regel werden Workflows danach charakterisiert, wie stark sie Abläufe à priori festlegen und welchen Flexibilitätsgrad sie während der Ausführung erlauben [Ser04].

Ad-hoc Workflow

Ad-hoc Workflows werden oft im Verwaltungsbereich eines Unternehmens eingesetzt und nach [Fis02] durch folgende vier Schritte charakterisiert:

1. Verhandlung – z. B. “Kannst du mein Dokument bis Freitag durchsehen?”
2. Annahme / Ablehnung – z. B. “Nein, aber wie wäre es bis Montag nachmittag?”
3. Zuweisung – die Arbeit an die Kritiker (engl. Reviewer) delegieren. Mehrere Personen können gleichzeitig ein Dokument durchsehen und sobald alle fertig sind, kann das Dokument weitergereicht werden – z. B. an das Management.
4. Review – wurde die ganze Arbeit erledigt und erfolgreich abgenommen? Falls nicht, wie sollen Ablehnungen oder unvollständige Bearbeitungen eines Dokumentes behandelt werden?

Diese Art von Workflow ist enorm flexibel und sorgt für eine gute Kontrolle des Prozesses: Wie ist der Status eines jeden Arbeitsschrittes? Wer hat wann was gemacht? Wo befindet sich die Arbeit gerade?

Gleichzeitig stellen Ad-hoc Workflows große Anforderungen an ein Workflow Management System. Es muss sehr flexibel sein und dynamische Änderungen eines Prozesses zur Laufzeit gewähren. Die Zustellung der Aufgaben innerhalb eines solchen Workflows erfolgt zumeist via E-Mail [Fis02].

Produktionsworkflow

Produktionsworkflows sind (stark) *strukturierte* Prozesse, deren Ablauf durch ihre Prozessdefinition fest vorgegeben ist [Kou95]. Beispiele dafür sind die Kreditvergabe, die Bearbeitung von Schadensansprüchen an Versicherungen oder die Buchhaltung – durchwegs (*hoch*) *strukturierte, transaktionsorientierte* Büroarbeiten. Im Laufe der letzten Jahre wurden derartige Produktionsorkflows so ausgebaut, dass sie zum zentralen Bestandteil der unternehmensweiten Datenverarbeitungssysteme wurden. Der größte Vorteil von Workflow in Produktionsanwendungen besteht in der gesteigerten betrieblichen Produktivität und in der verbesserten Behandlung von Ausnahmen [Por02]. Gewöhnlich wird für die Verteilung der Arbeit ein dedizierter Kanal verwendet, im Gegenteil zur Zustellung via E-Mail [Fis02].

Administrationsworkflow

In der Literatur wird manchmal der sogenannte *Administrationsworkflow* aufgelistet. Es ist eine Art Mischung aus Ad hoc- und Produktionsworkflow. Der Ablauf ist vordefiniert, kann aber im voraus geprüft oder sogar geändert werden. Die Zustellung der Aufgaben an die Workflow Teilnehmer erfolgt entweder via E-Mail oder einen kundenspezifischen Zustellungsmechanismus [Fis02]. Abbildung 4.2 zeigt noch einmal die vorgestellten Workflowtypen.

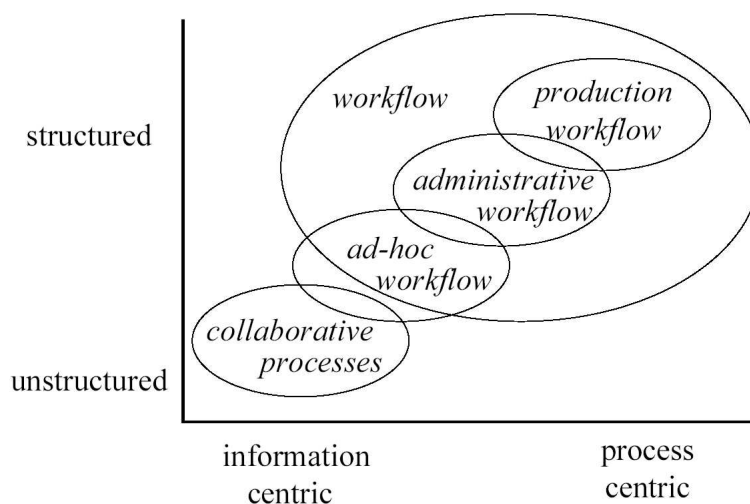


Abbildung 4.2: Arten von Workflow (Quelle:[vdA98])

Horizontal vs. Vertikal

Neben den vorgestellten Workflowtypen gibt es eine weitere Aufteilung der Workflows in *horizontale vs. vertikale Workflows*.

Ein *horizontaler Workflow* bewegt die Arbeit innerhalb eines Unternehmens von Person zu Person, zu verschiedenen Abteilungen oder Systemen. Man bezeichnet so einen Workflow auch als "Routing" [Fis02].

Der *vertikale Workflow* koordiniert die Weiterverarbeitung in jedem Schritt eines Prozessablaufs. Er ruft automatisch Computerprogramme auf, gibt Daten ein (z. B. Kontonummer) und bietet eine Anleitung in jedem Prozessschritt – insbesondere für Anfänger [Fis02].

4.3.4 Klassifikation von Workflow Systemen

Ähnlich der Kategorisierung verschiedener Workflows werden auch ganze *Workflow Systeme* hauptsächlich in zwei Gruppen klassifiziert.

Eingebettet (embedded)

Eingebettete Workflow Systeme sind leichtgewichtige Komponenten, die innerhalb einer umgebenden Applikation, z. B. eines Enterprise Portals, eingesetzt werden und dort die Abläufe in der umgebenden Applikation steuern [RT04]. Es werden keine speziellen Benutzerschnittstellen benötigt. Endbenutzer treten nicht direkt mit dem Workflow System in Kontakt, da ein vorgegebener Workflow fest in der umgebenden Applikation verankert ist [Fis02].

Autonom (stand-alone)

Im Gegensatz zu eingebetteten Workflow Systemen sind *autonome* Workflow Systeme umfangreiche und eigenständige Softwaresysteme, die Abläufe über verschiedene Anwendungen hinweg steuern [RT04]. Während der Modellierung eines Geschäftsprozesses müssen daher Schnittstellendefinitionen für externe Anwendungen erstellt werden, wodurch erheblicher Aufwand entstehen kann.

In der Regel haben autonome Workflow Systeme eigene Benutzerschnittstellen und greifen auf die Daten von anderen Anwendungen zu [zMA00].

4.4 Standardisierung: Die Workflow Management Coalition

Das Konzept des Workflow Managements wurde bereits Anfang der 80er erforscht und mit den Bemühungen mehrerer Arbeitskreise entwickelt. Auf diese Weise sind Workflow Produkte aus den verschiedensten Branchen entstanden. Einige dieser Produkte sind von Grund auf als Workflow Systeme entwickelt worden, viele entstanden aus Bildverarbeitungs-, Dokumentenmanagement-, Datenbank- und E-Mail-Systemen. Als Folge dieser Entwicklung sind zahlreiche Workflow Systeme kreiert worden, die jeweils nur auf einige wenige Aspekte des Workflow Managements spezialisiert waren. Obwohl Benutzer dadurch in der Lage waren, ein Produkt auszuwählen, das ihren Anforderungen und Ansprüchen genüge, fehlte eine gemeinsame Terminologie, um die Kooperation mehrerer Workflow Produkte zu unterstützen. Nur durch eine erfolgreiche Standardisierung würde es möglich sein, die Stärken verschiedener Workflow Systeme in einer Infrastruktur zu vereinen.

4.4.1 Die Workflow Management Coalition

Auf Grund der genannten Forderungen ist die *Workflow Management Coalition* – im Folgenden nur noch mit WfMC bezeichnet – im August 1993 als gemeinnützige, internationale Organisation von Workflow Anbietern, Benutzern und Analysten ins Leben gerufen worden [Coa04]. Obwohl die vorhandenen Workflow Produkte auf verschiedene Aspekte des Workflow Managements spezialisiert waren, hatten sie doch eine gemeinsame Charakteristik. Dies half der WfMC für manche Funktionen Standards zu definieren, um mehr Interoperabilität zwischen den verschiedenartigen Systemen zu erzielen.

Neben der Interoperabilität ist die Schaffung einer einheitlichen Terminologie für den Bereich Workflow Management ein wesentliches Ziel der WfMC. Dazu entwickelte die WfMC ein Referenzmodell für Workflow Management Systeme. Es legt allgemeine Eigenschaften der Systeme, deren funktionalen Bereiche und die benötigten Schnittstellen zwischen diesen Bereichen fest. Dabei enthält das Modell nur wenige Angaben zum internen Aufbau der Systeme und konzentriert sich vielmehr auf die Standardisierung der Schnittstellen [Coa95].

Die WfMC selbst sieht ihre Mission nach [Coa04] in der :

- Erhöhung des Wertes von Kundeninvestitionen durch Workflow Technologie

- Minimierung des Risikos, ein Workflow Produkt einzusetzen
- Expansion des Workflowmarkts durch Bekanntmachung von Workflow

4.4.2 Funktionen eines Workflow Management Systems

Generell lässt sich der Funktionsumfang eines Workflow Management Systems in 2 grobe Bereiche unterteilen, den *Build-Time* und *Run-Time* Bereich. Abbildung 4.3 zeigt die wesentlichen Zusammenhänge.

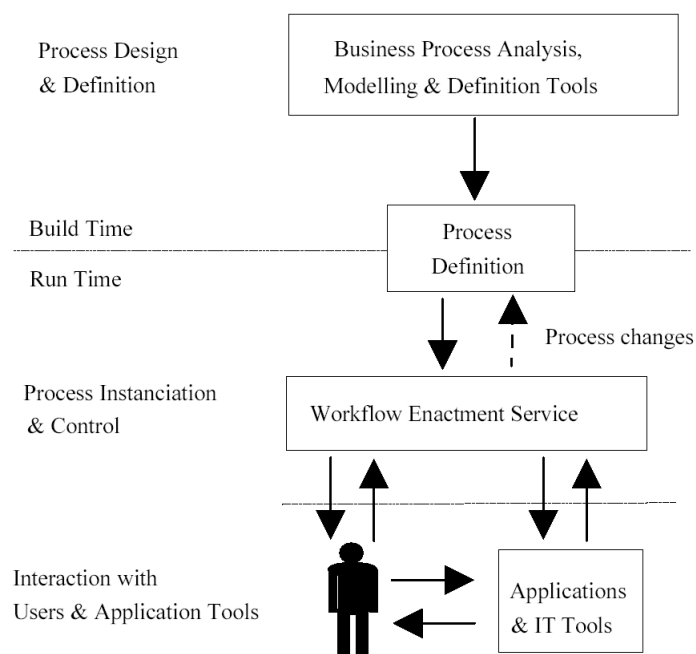


Abbildung 4.3: Merkmale eines Workflow Systems (Quelle:[Coa95])

Build-Time

Im Build-Time Bereich werden Prozesse definiert, die anschließend vom Workflow Enactment Service (dazu in Abschnitt 4.4.3 mehr) interpretiert und ausgeführt werden. Die Prozessdefinition muss dabei alle notwendigen Informationen enthalten, wie z. B. Start- und Endbedingungen, um eine korrekte Ausführung des Prozesses zu ermöglichen. Einige wenige Workflow Systeme haben die Fähigkeit, die Prozessdefinition während der Laufzeit zu modifizieren, was durch den gestrichelten Pfeil in Abbildung 4.3 deutlich gemacht wird [Coa95].

Run-Time

Die in der Build-Time entstandene Prozessdefinition wird in der Run-Time zur Ausführung gebracht. Dazu wird eine Prozessinstanz erzeugt und die Interaktion mit den Benutzern und Workflow Anwendungen kontrolliert [Coa95].

4.4.3 Generische Struktur von Workflow Management Systemen

Trotz der Unterschiede vorhandener Workflow Systeme, gibt es grundlegende Strukturen, die allen Systemen gemein sind. Aus diesem Grund hat die WfMC ein allgemeines Modell entworfen, das den Aufbau der meisten auf dem Markt befindlichen Workflow Produkte beschreibt (s. Abbildung 4.4). Es legt die wichtigsten funktionalen Komponenten eines Workflow Management Systems und die Schnittstellen zwischen ihnen fest.

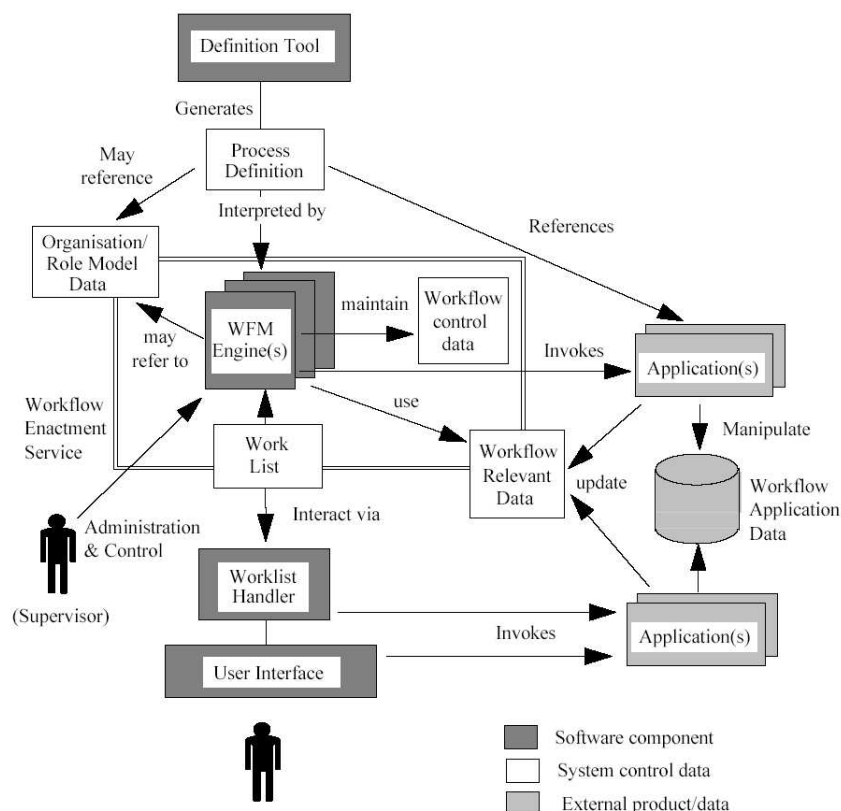


Abbildung 4.4: Generische Struktur von Workflow Produkten (Quelle:[Coa95])

Nach [Coa95] werden die einzelnen Komponenten in drei Klassen aufgeteilt:

- Softwarekomponenten, die grundlegende Funktionen innerhalb eines Workflow Management Systems zur Verfügung stellen (dunkelgrau).
- Verschiedene Arten von Prozessdefinitionen und Steuerdaten, die von einer oder mehreren Softwarekomponenten genutzt werden (weiß).
- Externe Anwendungen und Daten, die außerhalb des Workflow Systems liegen, aber als Teil des Gesamtsystems aufgerufen werden können (hellgrau).

Als nächstes werden die wichtigsten Komponenten des allgemeinen Modells erläutert.

Werkzeug zur Prozessdefinition (Process Definition Tool)

Das *Process Definition Tool* ermöglicht die Transformation eines Prozesses aus der realen Welt in eine Darstellung, die von der Workflow Engine interpretiert werden kann. Diese Darstellung kann entweder in textueller oder graphischer Form vorliegen [Coa95].

Weiterhin kann das *Process Definition Tool* Teil eines Workflow Produktes sein oder ein unabhängiges Werkzeug eines anderen Anbieters. Im zweiten Fall muss ein kompatibles Austauschformat in beiden Systemen implementiert sein.

Prozessbearbeitungsservice (Workflow Enactment Service)

Der *Workflow Enactment Service* besteht aus einer oder mehreren Workflow Engines und ist für die Erstellung, Verwaltung und Ausführung einzelner Prozessinstanzen verantwortlich [Coa99a]. Eine Workflow Engine übernimmt dabei je eine Prozessinstanz.

Der Aufgabenbereich eines Workflow Enactment Services umfasst die Interpretation der Prozessdefinitionen, die Kontrolle der Prozessinstanzen, die Sequenzierung in Beziehung stehender Aktivitäten, das Hinzufügen anstehender Elementaraufgaben in die Arbeitsliste eines Benutzers und den Aufruf von Anwendungen. Darüberhinaus verwaltet ein Workflow Enactment Service interne Steuerdaten, wie z. B. aktuelle Zustandsinformationen von Prozess- und Aktivitätsinstanzen [Coa95].

Ein wichtiges Merkmal ist der Bezug zum Unternehmensmodell. Enthält eine Prozessdefinition Referenzen auf bestimmte Rollen oder Strukturen eines Unternehmens, so muss der Workflow Enactment Service in der Lage sein, diese Infor-

mationen mit den beteiligten Benutzern während der Ausführung eines Prozesses zu verbinden.

Arbeitsliste (Worklist)

Überall dort, wo eine Benutzerinteraktion während der Prozessausführung notwendig ist, fügt die Workflow Engine Elementaraufgaben der *Arbeitsliste* des beteiligten Benutzers hinzu [Coa95]. In manchen Systemen ist dieser Vorgang unsichtbar, so dass Benutzer immer nur die nächste Aufgabe zu sehen bekommen. In einigen anderen Systemen hingegen können Benutzer die gesamte Arbeitsliste einsehen und anstehende Aufgaben in individueller Reihenfolge abarbeiten.

Worklist Handler & Benutzerschnittstelle (Workflow Client Applikation)

Der *Worklist Handler* eines Workflow Systems ist für die Interaktion des Workflow Enactment Services mit den Workflow Teilnehmern verantwortlich. Er unterstützt die Ausführung der Prozesse bei der Verwaltung von Aktivitäten, die auf Eingaben von Benutzern angewiesen sind.

Zu den Aufgaben eines Worklist Handlers gehört neben der Verteilung von Elementaraufgaben auf Benutzer auch das An- und Abmelden von Workflow Teilnehmern, das Anfragen von Elementaraufgaben, die für einen bestimmten Benutzer anstehen, etc...[Coa95] Wegen dem weiten Aufgabenbereich bevorzugt die WfMC für den Ausdruck *Worklist Handler* auch den Begriff *Workflow Client Applikation*.

Die *Benutzerschnittstelle* ist in Abbildung 4.4 zwar als separate Komponente dargestellt, verschmilzt in manchen Workflow Systemen aber auch mit dem Worklist Handler zu einer Softwarekomponente. Beide Komponenten können Anwendungen aufrufen, um Benutzer bei einer bestimmten Aufgabe zu unterstützen. Zum Beispiel könnte es erforderlich sein, dass jedesmal, wenn der Arbeitnehmer eine Aktivität beendet, eine Bestätigung automatisch an den Abteilungsleiter geschickt wird.

Workflow Steuerdaten, Workflowrelevante Daten und Anwendungsdaten

Workflow Steuerdaten werden innerhalb des Workflow Management Systems verwaltet und sind von außen normalerweise nicht zugänglich. In der Regel enthalten sie Informationen über den Status von Prozess- bzw. Aktivitätsinstanzen.

Daten, die von Workflow Anwendungen generiert und aktualisiert werden, heißen *Workflowrelevante Daten*. Im Gegensatz zu den Steuerdaten können workflowrelevante Daten von außen manipuliert werden und so den Status einer Prozessinstanz ändern. Anhand dieser Änderungen kann der Workflow Enactment Service seinerseits Entscheidungen für den weiteren Prozessverlauf treffen. Mit Vor- und Nachbedingungen, Übergangsbedingungen von einer Aktivität zur nächsten und der Zuweisung von Workflow Teilnehmern werden Zustandsübergänge einer Prozessinstanz bestimmt [Coa99a].

Anwendungsdaten gehören zu einer Workflow Anwendung (invoked application) und stehen dem Workflow Management System nicht zur Verfügung.

4.4.4 Das Referenzmodell der WfMC

Aus der generischen Struktur eines Workflow Management Systems entwickelte die WfMC ein Referenzmodell, indem sie Schnittstellen festlegte, die für die Interoperabilität verschiedener Systeme notwendig sind. Abbildung 4.5 illustriert das Referenzmodell, welches im Wesentlichen aus einzelnen Komponenten und Schnittstellen zwischen ihnen besteht.

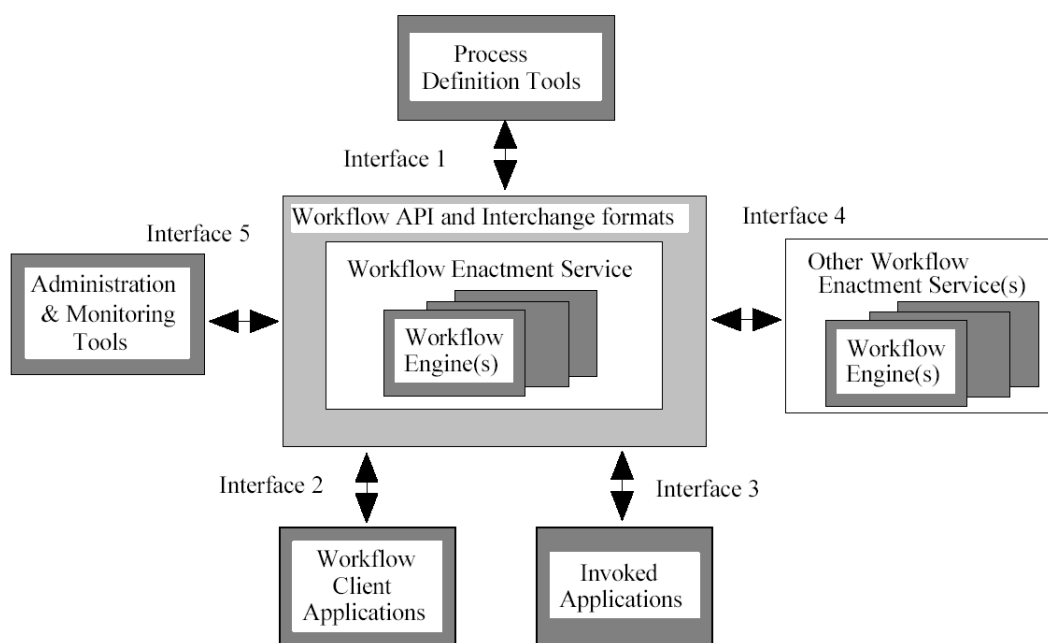


Abbildung 4.5: Das Workflow Referenzmodell (Quelle:[Coa95])

Die fünf Schnittstellen zum Workflow Enactment Service werden zusammen als WAPI bezeichnet – Workflow Application Programming Interfaces und Inter-

change Formats. Im Folgenden werden die einzelnen Schnittstellen nach [Coa95] genauer beschrieben.

Interface 1: Prozessdefinition und Austausch

Das Interface 1 des Referenzmodells besteht aus einer Menge von Funktionen und ist für den Austausch von Prozessdefinitionen zuständig. Es ermöglicht die Modellierungssoftware (Process Definition Tool) und die Laufzeitumgebung (Workflow Enactment Service) unabhängig voneinander zu wählen. Dadurch können zum einen Produkte eingesetzt werden, die für die jeweilige Aufgabe optimiert sind. Zum anderen können die gleichen Prozessdefinitionen von unterschiedlichen Workflowprodukten genutzt werden, was Voraussetzung für den Einsatz einer verteilten Laufzeitumgebung zur Lösung umfangreicher Aufgaben ist [Coa95].

Als Austauschformat wird von der WfMC die XML Process Definition Language (XPDL) [Coa02] propagiert. Die genaueren Details zu diesem Format werden im nächsten Kapitel in Abschnitt 5.1.1 vorgestellt.

Interface 2: Client-Anwendungen

Die Interaktion zwischen den Client-Anwendungen und dem Workflow Enactment Service findet durch das Interface 2 statt, das meistens als Arbeitsliste (engl. Worklist) organisiert ist. Diese Liste enthält Aufgaben, die von einem bestimmten Benutzer oder einer Gruppe von Benutzern zu erledigen sind.

Falls während der Ausführung einer Prozessinstanz Eingaben vom Benutzer erforderlich sind, fügt die Workflow Engine neue Einträge der Arbeitsliste hinzu. Diese werden von der Client-Anwendung (Worklist Handler) ausgelesen und dem Benutzer zur Abarbeitung bereitgestellt. Daneben gibt es weitere Funktionen, die das Entfernen erledigter Aufgaben, das Markieren von zwischenzeitlich nicht ausführbaren Aufgaben usw. ermöglichen.

Weiterhin kann durch das Interface 2 der Worklist Handler von dem Workflow Enactment Service entkoppelt werden. Dadurch kann eine an unternehmensspezifische Anforderungen angepasste Client-Anwendung mit verschiedenen Workflow Management Produkten interagieren – vorausgesetzt sie halten sich an die Schnittstellen des Referenzmodells [Coa95].

Interface 3: Aufgerufene Applikationen

Aufgerufene Applikationen sind Anwendungen, die vom Workflow Management System aufgerufen werden, um eine Aktivität (teilweise) automatisiert auszuführen oder den Workflow Teilnehmer bei der Abarbeitung einer Aufgabe zu unterstützen [Coa99a]. Dabei ist es unmöglich mit allen Anwendungen direkt zu interagieren, weil sie prinzipiell auf allen möglichen Plattformen in allen Netzwerkumgebungen in Frage kommen.

Viele Workflowprodukte können aus diesem Grund nur mit Anwendungen eines bestimmten Typs interagieren, bei dem die Daten einfach übergeben werden können und ihre Struktur bekannt ist, z. B. Textverarbeitungsprogrammen. Eine andere Möglichkeit stellen standardisierte Austauschmechanismen oder sogenannte *Anwendungsagenten* (engl. Tool Agents) dar. Solche Agenten verbergen die Unterschiede der Anwendungen durch die Implementierung einer bestimmten Technologie. Einige sind in der Lage MS Windows DDE Befehle zu interpretieren, andere kommunizieren über Protokolle wie MS OLE, CORBA, SOAP, RMI, RPC, etc. [Coa98]. Darüberhinaus ist es möglich, *workflowfähige Anwendungen* zu entwickeln, die zur Interaktion mit dem Workflow Enactment Service eine standardisierte Schnittstelle (API) benutzen.

Interface 4: Interoperabilität

Workflow Interoperabilität ist die Fähigkeit von zwei oder mehreren Workflow Engines zu kommunizieren und kooperieren, um Prozessinstanzen verteilt auszuführen und zu koordinieren [Coa99b]. Das Interface 4 soll damit verschiedenen Workflow Management Systemen die Möglichkeit geben, über die eigene Systemgrenze hinweg Arbeitsvorgänge zu unterstützen. Über diese Schnittstelle können Aktivitäten und Teilprozesse gestartet, Prozessdefinitionen, workflowrelevante Daten sowie Steuer- und Statusinformationen ausgetauscht und die Ausführung von Prozess- und Aktivitätsinstanzen an bestimmten Stellen synchronisiert werden.

Dazu ist von der WfMC eine XML-basierte Sprache (Wf-XML) [Coa01] definiert worden, welche die in [Coa99b] spezifizierten Arten von Interoperabilität modelliert. Darüberhinaus befinden sich standardisierte Protokolle (ASAP⁵, AWSP⁶) in der Entwicklung, welche die Interaktion von Workflow Engines über das Internet mit Hilfe von SOAP ermöglichen sollen.

⁵Asynchronous Service Access Protocol

⁶Asynchronous Web Services Protocol

Da dieser Teil des Workflow Managements nicht Schwerpunkt dieser Studienarbeit ist, sei zur weiteren Vertiefung in diesem Gebiet auf die Working Drafts von Wf-XML 2.0/AWSP (10/2003) [SGP03] und ASAP (09/2003) [Ric03] verwiesen.

Interface 5: Administration und Überwachung

Interface 5 des Referenzmodells dient der Administration und Überwachung von Workflow Enactment Services. Durch eine Menge ausgewählter WAPI-Funktionen können verschiedene Workflow Enactment Services einheitlich kontrolliert und der Status aller Prozessinstanzen überblickt werden. Nach [Coa95] bietet diese Schnittstelle Funktionen für die Bereiche Benutzermanagement, Rollenmanagement, Aufzeichnung von Ereignissen, Ressourcenkontrolle und Management von Statusinformationen an.

5 Technologieanalyse

In diesem Kapitel werden vorhandene Implementationen der im Referenzmodell der WfMC propagierten Softwarekomponenten und Schnittstellen untersucht. Dabei geht es in erster Linie um die Analyse von Beschreibungssprachen für Prozessdefinitionen (Interface 1, Abschnitt 5.1) und verfügbare Workflow Engines (Abschnitt 5.2).

5.1 Prozessbeschreibungs- und Ausführungssprachen

In den letzten Jahren wurden mehrere Sprachen entwickelt, mit denen sich Workflows beschreiben und modellieren lassen können. Jedoch hat sich bislang kein einheitlicher Standard etabliert, auf welchen man bedenkenlos setzen könnte. In diesem Abschnitt werden daher zunächst einige Beschreibungssprachen genauer analysiert, um konkrete Aussagen über ihren Reifegrad sowie deren Einsatzfähigkeit treffen zu können.

5.1.1 XML Process Definition Language (XPDL)

Die *XML Process Definition Language (XPDL)* [Coa02] ist die von der WfMC entwickelte Sprache für das Interface 1 (Prozessdefinition und Austausch) des Referenzmodells. Sie unterstützt den Im- und Export von Prozessdefinitionen und sorgt so für den Austausch von Prozessdefinitionen zwischen unterschiedlichen Workflowprodukten. XPDL ist im Oktober 2002 in der Version 1.0 veröffentlicht worden und wird seitdem in vielen Systemen als Austauschformat für Prozessdefinitionen implementiert.

Metamodell

Die Spezifikation von XPDL basiert auf einem Metamodell, welches gemeinsame Objekte, ihre Beziehungen untereinander und Attribute innerhalb einer Prozessbeschreibung identifiziert. Darüberhinaus dient es zur Abdeckung der minimalen Anforderungen an eine vollständige Prozessdefinition. Abbildung 5.1 illustriert die einzelnen Entitäten und Beziehungen des XPDL Metamodells.

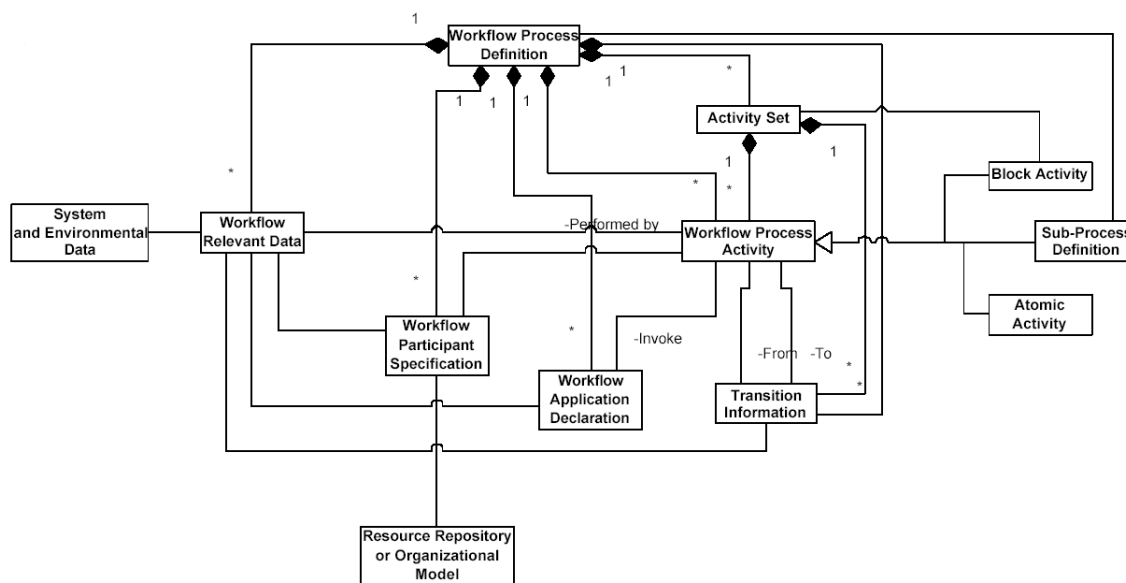


Abbildung 5.1: Metamodell von XPDL

Workflow Process Definition enthält kontextabhängige Informationen, die sich auf andere Entitäten innerhalb des beschriebenen Prozesses beziehen. Es ist ein Container für den Prozess selbst, welcher Informationen bezüglich der Administration (Erstellungsdatum, Autor, etc.) oder der Prozessausführung (Initiale Parameter, Priorität, Zeitschranken, zu benachrichtigende Person, etc.) zur Verfügung stellt.

Workflow Process Activity besteht aus einer oder mehreren Aktivitäten, von denen jede jeweils eine logische, in sich geschlossene Arbeitseinheit darstellt. Eine Aktivität repräsentiert Arbeit, die einem Workflow Teilnehmer und/oder einer Anwendung zugewiesen werden kann. Neben Informationen über die Art der Ausführung (*automatisiert/manuell*) können auch In- und Out-Parameter spezifiziert werden, die den Austausch von workflowrelevanten Daten erlauben.

Transitionen stellen die Verbindung zwischen Aktivitäten dar. Jede Transition besitzt dazu drei elementare Eigenschaften: **from**-Aktivität, **to**-Aktivität und die Bedingung, unter welcher von der einen zur nächsten Aktivität geschaltet wird, z. B.

```
<Transition Id="10" From="36" To="22">  
  <Condition Type="CONDITION">docStatus == "accepted"</Condition>  
</Transition>
```

Neben **CONDITION** gibt es zusätzlich die Typen **OTHERWISE** (Ausführung der Transition, falls die Bedingung nicht erfüllt wird), **EXCEPTION** (Ausführung der Transition, falls eine Ausnahme auftritt und die Bedingung erfüllt wird) und **DEFAULTEXCEPTION** (Ausführung der Transition im Falle einer Ausnahme, unabhängig von der Erfüllung der gestellten Bedingung).

Workflow Participant Specification beschreibt die an einem Workflow beteiligten Akteure. Dies können nicht nur Personen und Rollen eines Unternehmens sein, sondern auch beliebige Ressourcen, z. B. eine Datenquelle. In besonders anspruchsvollen Szenarien kann die *Workflow Participant Specification* ihre Informationen auch von einem *Resource Repository* beziehen oder im Falle von menschlichen Workflow Teilnehmern von einem *Unternehmensmodell*.

Workflow Application Declaration offeriert Beschreibungen von Anwendungen oder Schnittstellen, die vom Workflow Management System aufgerufen werden, um die Abarbeitung einer Aktivität zu unterstützen oder vollständig zu automatisieren. Die Beschreibung einer jeden Anwendung wird durch eine eindeutige ID gekennzeichnet und kann über ein Attribut mit einer Aktivität verknüpft werden.

Workflow Relevant Data wird innerhalb der Prozessdefinition deklariert und ist vom Prinzip her eine Variable eines bestimmten Datentyps. Neben zahlreichen Standardtypen, wie **String**, **Integer**, **Float**, **Referenz**, **Date/Time** können eigene Typen in Form einer *XML Schema Definition*¹ verwendet werden. Diese kann entweder im XPD L Dokument eingebettet sein oder über das **ExternalReference**-Element von einer externen Quelle eingebunden werden.

Die definierten Variablen werden bei der Instanziierung eines neuen Prozesses angelegt und stehen den Aktivitätsinstanzen oder Anwendungen zur

¹<http://www.w3.org/XML/Schema>

Verfügung. Sie übermitteln Informationen und Ergebnisse von einer Aktivität zur anderen und werden in bedingten Ausdrücken bei einer Transition ausgewertet oder für die Zuweisung einer Aktivität an einen Workflow Teilnehmer verwendet.

System and Environmental Data meint Daten, die vom Workflow Management System oder der lokalen Systemumgebung verwaltet werden. Sie können überall dort, wo workflowrelevante Daten zum Einsatz kommen, genutzt werden.

Packages

Eine besondere Eigenschaft von XPDL ist die Möglichkeit, Teilnehmer, Anwendungen, workflowrelevante Daten und sogar ganze Prozessdefinitionen zu einem *Package* zusammenzufassen. Einmal definierte Daten können dadurch in mehreren Prozessdefinitionen verwendet werden und müssen nicht jedesmal neu definiert werden.

Flexibilität und Erweiterbarkeit

XPDL ist darauf ausgelegt, bei Bedarf größtmögliche Flexibilität und Erweiterbarkeit implementieren zu können. Dies wird dadurch ermöglicht, dass innerhalb des standardisierten Teils von XPDL auch benutzerspezifische Daten mit Hilfe des Elements *ExtendedAttribute* definiert werden können, z. B.

```
<ExtendedAttributes>
  <ExtendedAttribute Name="SystemActivity" value="Notify">
  <ExtendedAttribute Name="Email">
    <xyz:Email to="%%editor.emailAddress" subject="%%document_reviewed">
      The document %%document has been reviewed.
      Everything was fine!
    </xyz:Email>
  </ExtendedAttribute>
</ExtendedAttributes>
```

Im Wesentlichen besteht so ein erweitertes Attribut aus einem Namen und Wert, der gegebenenfalls aus einem spezifischen Element bestehen und Verweise auf workflowrelevante Daten enthalten kann. Die definierten Attribute können in allen anderen Entitäten verwendet und referenziert werden.

5.1.2 WSFL, XLANG, BPEL4WS

Die *Web Services Flow Language (WSFL)* ist eine XML-basierte Auszeichnungssprache zur Beschreibung von *Web Service - Kompositionen*. Sie wurde von der IBM Software Group als Teil des Web Service Frameworks entwickelt und beruht auf bestehenden Spezifikationen wie SOAP, WSDL und UDDI. Die Spezifikation von WSFL [Ley01] wurde im Mai 2001 in der Version 1.0 veröffentlicht und orientiert sich stark am Petri-Netz-Modell.

WSFL ist ausschließlich in Kombination mit WSDL zu sehen, da WSFL auf die durch WSDL beschriebenen Web Services zugreifen soll. Mit diesem Ansatz können langlebige Geschäftsprozesse über die Grenzen eines Unternehmens hinaus realisiert werden.

Metamodell

Die Spezifikation von WSFL basiert auf dem Modell eines virtuellen Unternehmens, das im Internet Bücher verkauft. Dieses Unternehmen verschickt die Bücher allerdings nicht, sondern beauftragt einen Transporteur, die gewünschte Ware von einem Buchhändler abzuholen und sie dem Kunden zu überbringen. Die Kommunikation findet dabei über die im jeweiligen WSDL Dokument beschriebenen Schnittstellen statt. Abbildung 5.2 illustriert die essentiellen Bestandteile einer WSFL Prozessbeschreibung.

In dem Modell übernimmt *Role A* die Rolle eines Transportunternehmens, *Role B* steht für eine Internet-Buchhandlung. WSFL Aktivitäten (`<activity>`, als Kreise dargestellt) beschreiben einzelne Arbeitsschritte, die über `<controlLink>`-Elemente zu einem Workflow verknüpft werden. Weitere strukturelle Elemente des WSFL Modells werden in den folgenden Abschnitten genauer beschrieben.

Definition eines Workflows

Um einen Workflow in WSFL zu definieren benötigt man nur einige wenige Elemente:

flowSource definiert den Startpunkt eines Workflows.

serviceProvider stellen die am Workflow teilnehmenden Service Anbieter dar.

activity beschreibt eine einzelne Aktivität innerhalb des Workflows. Für die Ausführung ist ein durch das **serviceProvider**-Element definierte Service Anbieter zuständig.

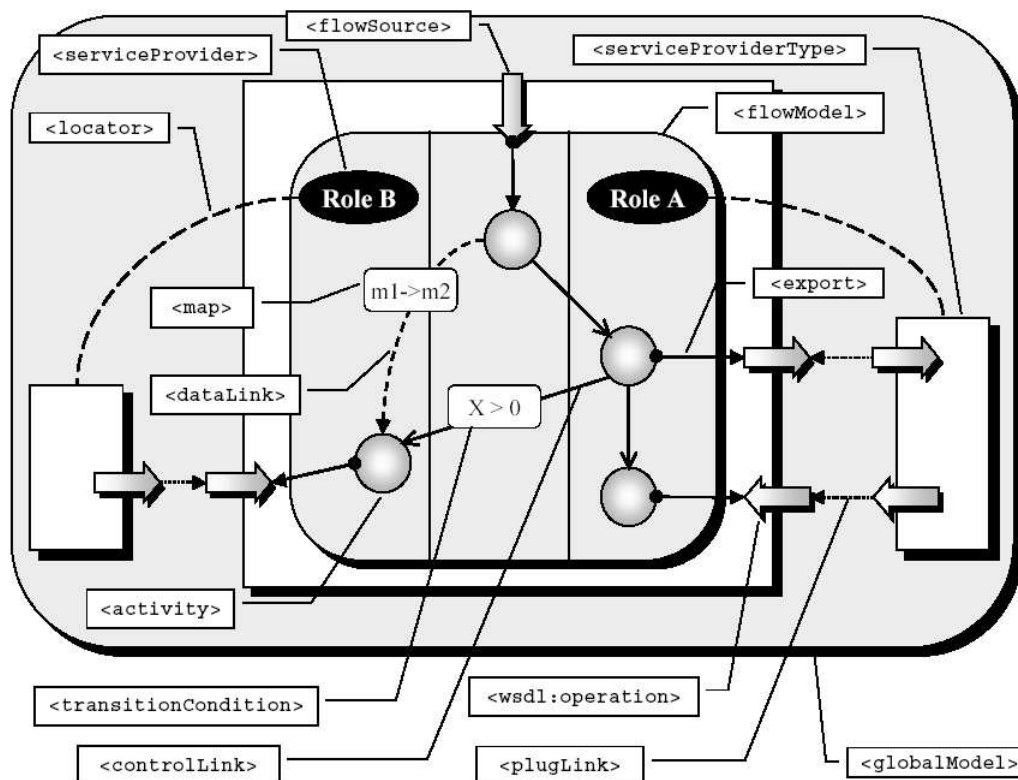


Abbildung 5.2: Ein WSFL Flow Modell (Quelle:[Ley01])

controlLink verbinden die Aktivitäten zum eigentlichen Workflow. Neben einfachen sequentiellen Verbindungen können auch komplexere Verzweigungen realisiert werden, beispielsweise die Synchronisation mehrerer eingehender **controlLinks** zu einer einzigen Operation.

Transitionen

Zwischen den einzelnen Aktivitäten und Web Services können in WSFL unterschiedliche Verbindungen existieren.

control links sind für den Kontrollfluss innerhalb des Workflows verantwortlich, vergleichbar mit dem Transition-Element in XPDL, z. B.

```
<controlLink source="processP0" target="acceptShipmentRequest"
  transitionCondition="pOutput.x > acceptSRInput.y"/>
```

Im Falle einer Ausnahme können bei Bedarf Fehlermeldungen innerhalb des **controlLink**- Elements erzeugt werden, die ein vom Anwender gewünschtes Verhalten erzwingen.

data links stellen den Datenfluss zwischen den unterschiedlichen Aktivitäten si-

cher. Sie werden analog zum `controlLink` mit der Angabe des Quell- und Zielknotens spezifiziert und enthalten in Allgemeinen einen Verweis auf eine im WSDL Dokument definierten Nachricht.

plug links: stellen Aufrufe von Web Services externer Service Provider dar.

Rekursive Komposition

Eine Besonderheit von WSFL ist die extensive Unterstützung *rekursiver Kompositionen*. Jede Web Service Komposition (WSFL Workflow) kann wiederum als Web Service betrachtet werden und somit als Service Anbieter in neuen Kompositionen verwendet werden. Mit diesem Ansatz wird eine gute Skalierbarkeit der Sprache erreicht und die Unterstützung für ein Top-Down Design gewährleistet [Ley01].

XLANG

XLANG [Tha01] ist ein von Microsoft avancierter Standard für den Nachrichtenaustausch zwischen Web Services. Ziel der Entwicklung war genau wie bei WSFL, die Automatisierung von Geschäftsprozessen über die Grenzen eines Unternehmens hinaus zu unterstützen. XLANG wurde im Mai 2001 der Öffentlichkeit zugänglich gemacht und ermöglicht seitdem, Web Services als Komponenten in langlebige Geschäftsprozesse einzubinden.

Im Gegensatz zu den bisher vorgestellten Sprachen ist XLANG keine eigenständige Sprache, sondern stellt lediglich eine Erweiterung von WSDL dar. XLANG spezifischen Erweiterungen eines WSDL Dokuments beschreiben das Verhalten eines Web Services. Das folgende Beispiel soll einen Eindruck der Gesamtstruktur einer XLANG Service Definition vermitteln.

```
<definitions>
  <portType name="RequestReceivePortType">
    <operation name="AskLastTradePrice">
      <input message="..."/>
    </operation>
  </portType>
  <portType name="ResponseSendPortType">
    <operation name="SendLastTradePrice">
      <output message="..."/>
    </operation>
  </portType>
```

```
...
<service name="StockQuoteProviderService">
...
  <port name="pGetRequest" ...> ... </port>
  <port name="pSendResponse" ...> ... </port>

  <xlang:behavior>
    <xlang:body>
      <xlang:sequence>
        <xlang:action operation="AskLastTradePrice"
          port="pGetRequest" activation="true"/>
        <xlang:action operation="SendLastTradePrice"
          port="pSendResponse"/>
      </xlang:sequence>
    </xlang:body>
  </xlang:behavior>

</service>
</definitions>
```

Das Beispiel enthält ein einfaches Verhalten, welches die Reihenfolge definiert, in der zwei asynchrone Operationen einer Preisanfrage ausgeführt werden. Im Allgemeinen lässt sich XLANG durch folgende Merkmale charakterisieren:

- Sequentielle und parallele Kontrollstrukturen
- Langlebige Transaktionen
- Benutzerspezifische Korrelation von Nachrichten
- Behandlung von Ausnahmen
- Modulare Beschreibung des Verhaltens von Web Services

Da XLANG nicht mehr weiter entwickelt wird, wird an dieser Stelle nicht tiefer auf einzelnen Sprachkonstrukte eingegangen.

BPEL4WS

Die *Business Process Execution Language for Web Services (BPEL4WS)* ist die Zusammenführung der Sprachen WSFL und XLANG. BPEL4WS spezifiziert das Verhalten von Geschäftsprozessen, die auf Web Services basieren, und hat WSFL

und XLANG mittlerweile abgelöst. Die aktuelle Spezifikation [ea03] wird vor allem von Microsoft und IBM vorangetrieben und liegt in der Version 1.1 vom Mai 2003 vor.

Die Spezifikation eines Geschäftsprozesses geschieht im Wesentlichen mit den folgenden Elementen:

- Vereinbarung einer *Ausführungsreihenfolge* für die zugehörigen Nachrichten einer Sammlung von Web Services
- Beschreibung der *Daten*, die zwischen teilnehmenden Web Services ausgetauscht werden
- Beschreibung der *Partner*, die an den Interaktionen teilnehmen, und deren *Rolle*
- Formulierung einer gemeinsamen *Ausnahmebehandlung*

Eine Prozessdefinition hat in BPEL4WS dann folgende Struktur [AE03]:

```
<process name="Reisebuero" xmlns="...>
  <partners>
    <partner name="... myRole="... />
  <partner name="... partnerRole="... />
  </partners>
  <containers>
    <container name="... messageType="... />
  <container name="... messageType="... />
  </containers>
  ...

```

Zunächst werden die am Prozess beteiligten *Partner* sowie die internen Datenstrukturen (*containers*) aufgelistet, in denen eintreffende Nachrichten gespeichert oder aus denen ausgehende Nachrichten entnommen werden können. Die eigentliche Ablaufbeschreibung erfolgt dann so:

```
...
<sequence>
  <receive name="... partner="... portType="...
    operation="... container="... />
  <invoke name="... partner="... portType="... operation="...
    inputContainer="... outputContainer="... />

```

```

    <reply name="... partner="... portType="...
        operation="... container="... />
</sequence>
</process>

```

Die Sequenz beschreibt, dass zunächst eine Nachricht empfangen werden muss, worauf hin eine Operation eines Web Services aufgerufen wird. Anschließend wird eine Antwort an einen Partner geschickt. Die notwendigen Parameter stammen dabei jeweils aus den Containern.

5.1.3 WSCI

WSCI steht für *Web Service Choreography Interface* und ist der von Sun, BEA, SAP und Intallio vorangetriebene Ansatz zur Beschreibung von Geschäftsprozessen. Die gegenwärtige Spezifikation liegt in der Version 1.0 vom August 2002 vor [SBIS02]. Die Struktur von WSCI ist der von BPEL4WS sehr ähnlich. Aus diesem Grund wird auf WSCI nicht tiefer eingegangen sondern kurz anhand eines Beispiels veranschaulicht.

Man betrachte einen Prozess im Reisebüro, wenn es eine Buchungsanfrage vom Kunden erhält. Zunächst soll das Reisebüro eine Reiseanfrage erhalten und anschließend eine Buchung für Tickets. Daraufhin bucht das Reisebüro die Tickets bei einer Flugesellschaft und leitet nach Erhalt der Bestätigung diese an den Kunden weiter. In WSCI sieht diese Interaktion wie folgt aus:

```

<interface name="Reisebuero">
  <process name="bucheReise" instantiation="message">
    <sequence>
      <action name="initiiereReisebuchung" role="tns:reisebuero"
        operation="tns:bucheReise"/>
      <action name="empfangenTicketanfrage" role="tns:reisebuero"
        operation="tns:bucheFlug">
        <call process="tns:bucheSitze"/>
      </action>
      <action name="sendeBestaetigung" role="tns:reisebuero"
        operation="tns:bestaetigung"/>
    </sequence>
  </process>
  ...
  <process name="bucheSitze"...
</interface>

```

Das `interface`-Element enthält mehrere Prozessdefinitionen, von denen der erste „`bucheReise`“ heißt und durch den Empfang einer Nachricht gestartet wird. Der Prozess enthält eine Abfolge von drei Aktionen: `bucheReise`-Operation, `bucheFlug`-Operation, die zum Aufruf eines weiteren Prozesses führt und die `bestaetige`-Operation, in welcher der Kunde über den Erfolg der Buchung informiert wird. Zur Vervollständigung bleibt der zweite Prozess `bucheSitze`, der vom ersten aufgerufen wird.

```
<process name="bucheSitze" instantiation="other">
  <action name="bucheSitze" role="tns:reisebuero"
    operation="tns:bucheSitze"/>
</action>
</process>
```

Neben `sequence` existieren natürlich noch weitere Elemente zur Workflowmodellierung, z. B. `choice`, `foreach`, `while`, etc...

5.1.4 Business Process Modelling Language (BPML)

Diese Workflowbeschreibungssprache BPML[Ark02] ist im November 2002 von der Business Process Management Initiative (BPMI²) in der letzten Version veröffentlicht worden und basiert ebenfalls auf XML. Generell basiert BPML auf endlichen Automaten, mit denen man das Verhalten von Prozessen und deren Interaktion beschreiben kann. Das Modell enthält Aktivitäten unterschiedlicher Komplexität, Transaktionen und ihre Kompensation, Datenmanagement, Nebenläufigkeit und Ausnahmebehandlung.

Aktivitäten

Im Vergleich zu anderen Sprachen gibt es in BPML eine Vielzahl von Aktivitätstypen. Bei Bedarf kann eine Aktivität – analog zu Subprozessen – in rekursiv definierte Subaktivitäten aufgeteilt werden. Abbildung 5.3 zeigt die unterschiedlichen Aktivitätstypen.

Grundsätzlich wird zwischen *komplexen* und *simplen* Aktivitäten unterschieden. Eine simple Aktivität besteht aus einer atomaren Aktion und lässt sich nicht weiter unterteilen. Eine komplexe Aktivität hingegen setzt sich aus mehreren Subaktivitäten zusammen: entweder aus simplen oder rekursiv definierten komplexen Aktivitäten. Die Funktion der in Abbildung 5.3 identifizierten simplen Aktivitäten ist in Tabelle 5.1 zusammengefasst.

²<http://www.bpmi.org>

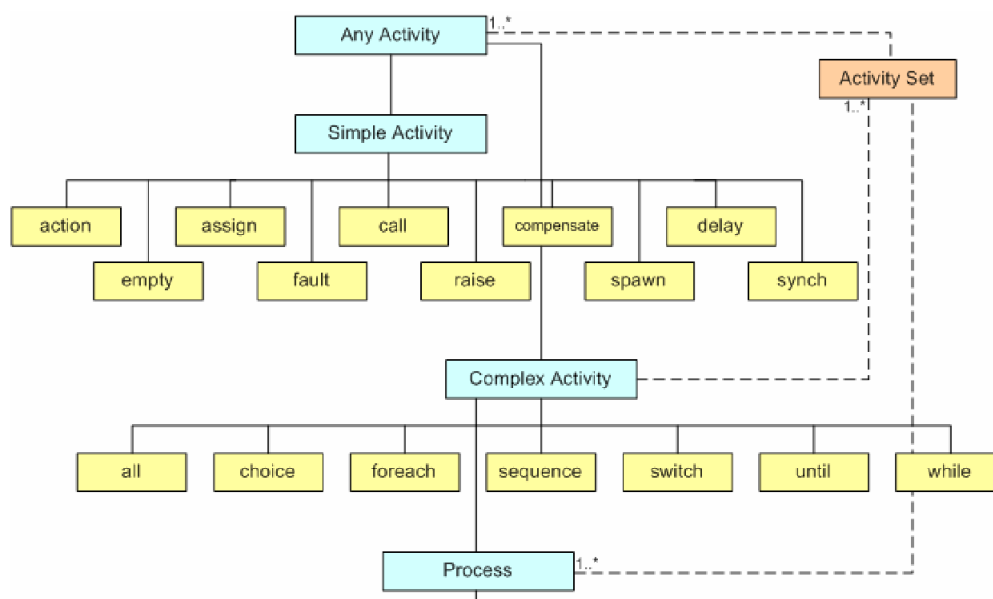


Abbildung 5.3: Aktivitätstypen in BPML (Quelle:[Ark02])

Eine Besonderheit von BPML ist die Verwendung von *Signalen*. Sie werden dazu verwendet, die Ausführung von Aktivitäten zu koordinieren, z. B. den Start einer Aktivität mit dem Ende einer anderen zu synchronisieren. Die **raise** und **synch** Aktivitäten tauschen Informationen über eine Signalinstanz aus.

Komplexe Aktivitäten entstehen durch Anwendung der von imperativen Programmiersprachen bekannten Konstrukte auf beliebige Aktivitäten (*Any Activity*, simple oder komplexe Aktivität). Einzige Besonderheit stellt die **all**-Aktivität dar. Sie bedeutet, dass alle innerhalb der **all** enthaltenen Aktivitäten parallel ausgeführt werden.

Definition und Instanziierung eines Prozess

Wie der Abbildung 5.3 zu entnehmen ist, wird unter einem Prozess eine komplexe Aktivität verstanden. Neben einfachen Prozessen spielen die so genannten Ausnahme- und Kompensationsprozesse. Sie sind dazu da, um im Falle einer Ausnahme die Änderungen eines Prozesses rückgängig zu machen und evtl. beanspruchte Ressourcen wieder freizugeben. Die Instanziierung kann auf drei unterschiedliche Arten erfolgen:

1. Durch ein Signal einer beliebigen Aktivität
2. Sobald eine bestimmte Nachricht vorliegt

Simple Aktivität	Funktion
action	Führt eine Operation durch zum Austausch von Eingangs- und Ausgangsnachrichten
assign	Weist einer Variable (property) einen neuen Wert zu
call	Instanziert einen Prozess und wartet auf dessen Terminierung
compensate	Kompensiert einen Prozess
delay	Wartet für eine gegebene Zeit
empty	Tut nichts. Wird dort eingesetzt, wo eine Aktivität erwartet wird, aber keine Arbeit zu verrichten ist
fault	Erzeugt einen Fehler im aktuellen Context
raise	Erzeugt ein Signal, welches zur Synchronisation von Prozessen verwendet wird
spawn	Instanziert einen Prozess und wartet <i>nicht</i> auf dessen Terminierung
synch	Synchronisiert ein Signal

Tabelle 5.1: Simple Aktivitäten in BPML

3. Direkt durch eine Aktivität oder anhand eines Zeitplans

5.1.5 Business Process Modelling Notation (BPMN)

BPMN ist eine UML ähnliche Modellierungssprache zur Beschreibung von Geschäftsprozessen. Im Gegensatz zu UML definiert sie jedoch nur einen einzigen Diagrammtyp, der auf dem so genannten *Process Execution Meta-Model* basiert. Die Spezifikation von BPMN ist von der BPMI im August 2003 veröffentlicht worden und wird im Juli 2004 in der finalen Version 1.0 erwartet.

Motivation und Zielsetzung

Die Motivation für BPMN ist eine gemeinsame Notation für die Modellierung von Geschäftsprozessen. Inkompatible Modellierungen ein und des selben Prozesses haben in der Vergangenheit oft zu Verwirrungen und Missverständnissen zwischen den Entwicklern und Endanwendern geführt. Eine eindeutige sowohl für Geschäftsanalysten als auch Entwickler verständliche Visualisierung der Geschäftsprozesse war von Nöten [dbN04].

Modellierung und Translation

Als erster Anbieter hat ITpearls³ ein Modellierungswerkzeug als MS Visio Plugin für BPMN entwickelt. Mit Hilfe dieses Werkzeugs können Prozesse in Form eines BPMN-Diagramms erstellt werden und dann nach BPML oder BPEL4WS übersetzt werden. Dies wird durch so genannte *Forward*-Translatoren erreicht. Die erzeugte Prozessdefinition kann anschließend von einem Workflow oder Business Process Management System importiert und weiterverarbeitet werden. Abbildung 5.4 zeigt den prinzipiellen Entwicklungszyklus.

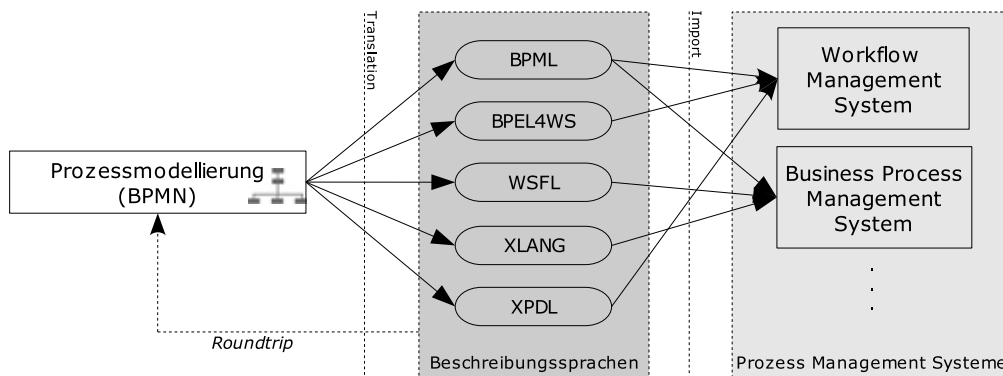


Abbildung 5.4: Translation von BPMN Diagrammen

Für die Zukunft ist die Entwicklung von *Roundtrip*-Translatoren gedacht [ITp03]. Sie sollen die Rücktransformation einer Prozessdefinition in einer beliebigen Beschreibungssprache in ein BPMN-Diagramm ermöglichen. Dieser Ansatz ist vergleichbar mit verschiedenen Roundtrip-Engineeringtools, wie z. B. von UML zum Quellcode und wieder zurück.

5.2 Workflow Engines

Neben einer Beschreibungssprache ist eine Workflow Engine für die Implementierung einer Workflowkomponente von eminenter Wichtigkeit. In diesem Abschnitt werden daher einige der interessantesten Open Source Engines vorgestellt.

5.2.1 Java Business Process Management (jBpm)

jBpm⁴ ist eine flexible und mächtige Laufzeitumgebung für die Ausführung von Geschäftsprozessen. Es orientiert sich stark am Referenzmodell der WfMC und

³<http://www.itpearls.com>

⁴<http://www.jbpm.org>

ist beim Prozessentwurf auf die Einbindung menschlicher Interaktion ausgerichtet. jBpm verwaltet den Zustand eines jeden Prozesses, logt jegliche Änderung und führt alle automatisierten Aktionen aus. Darüberhinaus kombiniert jBpm die einfache Entwicklung von Workflow Anwendungen mit ausgezeichneten EAI Fähigkeiten.

Die Beschreibung der Geschäftsprozesse erfolgt in der eigens dafür entwickelten *jBpm Process definition language (jPdl)*. Im Vergleich zu den bisher vorgestellten Sprachen ist jPdl wesentlich flexibler, bietet dem Entwickler intelligente Erweiterungsmöglichkeiten, kann einfach in bestehende Applikationen integriert werden und orientiert sich an den Workflow Patterns von WIL VAN DER AALST [vdABtHK00, vdA03]. Die enorme Flexibilität der Sprache wird dadurch erreicht, dass die *jBpm-Engine* nur für die Ausführung der Basisstruktur eines in jPdl definierten Prozesses zuständig ist. Jede nur denkbare variable Geschäftslogik wird zu einer Menge von Schnittstellen zusammengefaßt und somit entkoppelt.

Mit diesem Konzept ist es dem Entwickler nun möglich, die bestehende Default-Implementierung durch seine eigene – in Form einer Java-Klasse, die das entsprechende Interface implementiert – zu ersetzen. Zum besseren Verständnis betrachte man folgenden Auszug einer Prozessbeschreibung:

```
<process-definition>
...
<activity-state name="evaluating">
  ...
  <field attribute="evaluation_result" access="write-only" />
  <transition to="evaluation" />
</activity-state>

<decision name="evaluation"
  handler="org.jbpm.workflow.delegation.impl.decision.EvaluationDecision">
  <parameter name="attribute">evaluation result</parameter>
  <transition name="approve" to="approved_holiday_fork" />
  <transition name="disapprove" to="disapproval_notification" />
</decision>
...
</process-definition>
```

Ein graphischer Designer ist ebenfalls vorhanden (jgpd⁵), der Prozessdefinitionen nach jPdl exportiert.

⁵<http://sourceforge.net/projects/jgpd>

5.2.2 Enhydra Shark

*Enhydra Shark*⁶ ist ein Workflow Framework, das vollständig auf den Spezifikationen der WfMC basiert und XPDL (ohne spezifische Erweiterungen) als Beschreibungsformat für die Prozessdefinitionen verwendet. Herausragendes Merkmal von Shark ist die modulare Architektur mit vielen Erweiterungsmöglichkeiten für projektspezifische Anforderungen. Jede einzelne Komponente oder API (User Management, Persistenzschicht, Prozess Repository, Zuweisungs-, Benachrichtigungs-, Eskalations-APIs) kann mit der Standardimplementierung genutzt werden oder durch eine eigene Implementierung erweitert bzw. ersetzt werden.

Ein weiterer Vorteil von Shark ist die Zusammenarbeit mit dem *JaWE*⁷ Workflow Editor, welcher die erstellten Prozessdefinitionen im XPDL-Format exportiert. Durch das Zusammenspiel ist es möglich, Prozessinstanzen während der Ausführung zu visualisieren und dynamisch zu verändern. Letzteres ist noch in der Entwicklungsphase, da die WDF (Workflow Definition Functions) API zum jetzigen Zeitpunkt von der WfMC noch nicht verabschiedet worden ist.

5.2.3 Bonita

Bonita⁸ ist ein kooperatives Workflow System, das den Arbeitsfluss in einem Unternehmen spezifiziert, ausführt, überwacht und koordiniert. Das Gesamtsystem besteht im Wesentlichen aus zwei Komponenten: Modellierung und Ausführung.

Modellierung

Mit dem *GraphEditor* können Benutzer Prozesse graphisch definieren und editieren. Desweiteren können verschiedene Attribute einer Aktivität gesetzt werden (Typ, Beschreibung, Ausführungsmodus, Deadline, Rolle, ...) und Aktionen und/oder Eigenschaften assoziiert werden. Als besonderes Feature ist die Definition und Visualisierung durch unterschiedliche Benutzer erlaubt.

Ausführung

Die Ausführungskomponente erlaubt es Benutzern, die Prozessausführung zu kontrollieren. Außerdem ist bereits eine Worklist Anwendung integriert, die aus einer Projekt-, Todo-, und Aktivitätenliste besteht. Durch die Implementierung

⁶<http://shark.objectweb.org/>

⁷<http://jawe.objectweb.org/>

⁸<http://bonita.objectweb.org>

von Web Services kann eine benutzerspezifische Präsentationsschicht generiert werden.

5.2.4 Sonstige

Neben den angeführten drei Workflow Engines existieren noch viele andere, auf die hier aber nicht tiefer eingegangen wird. Eine regelmäßig aktualisierte Übersicht findet sich unter <http://www.topicus.nl/topwfm/Tooloverzicht.htm> [13.09.2004].

6 Evaluierung

Nach der Vorstellung der aktuellen Technologien im Bereich der Workflowmodellierung und -ausführung, wird in diesem Kapitel zunächst VKC vorgestellt und desweiteren anhand der resultierenden Anforderungen eine Bewertung der Workflow Beschreibungssprachen vorgenommen.

6.1 Virtual Knowledge Center (VKC)

VKC¹ ist ein Dokumenten- und Wissensmanagementsystem, welches eine Basis für die Ablage von Dokumenten (Berichte, Statistiken, Gutachten, Patente, etc.) bietet und das Know-How einer Organisation gezielt verteilt. Informationen und Wissen werden einfach und ohne großen Aufwand über eine Web-Schnittstelle zugänglich gemacht.

Alle Dokumente werden zentral in einer Dokumentablage bereitgestellt und können in verschiedenen Kategorien klassifiziert werden. Wo immer Personen miteinander kooperieren, können sie auf die gemeinsame Datenbasis zugreifen und verweisen.

VKC ist als Java Web-Anwendung (JSP², Struts³) realisiert und läuft unter folgender Systemkonfiguration:

- Windows (ab NT4) und Unix-Systemen (Linux, Solaris)
- JDK 1.4 oder höher
- Unabhängig von der Web-Server-Plattform
- Interbase 6 oder Firebird 1 (OpenSource) und Oracle 9i

¹<http://www.vkc.info>

²Java Server Page

³Web-Framework, <http://jakarta.apache.org/struts/>

6.2 Bewertung von Workflow Beschreibungssprachen

In diesem Abschnitt werden die vorgestellten Workflow Beschreibungssprachen aus Kapitel 5.1 diskutiert und bezüglich der Einsatzfähigkeit im Dokumentenmanagementsystem VKC beurteilt. Es werden die Stärken und Schwächen der einzelnen Sprachen genannt und teilweise mit einander verglichen.

6.2.1 XPDL

XPDL ist der von der WfMC entwickelte Standard zur Beschreibung von Prozessdefinitionen.

Stärken

Die Stärken und Besonderheiten von XPDL sind:

- **Ressource Repository**

Um Redundanz zu vermeiden, werden workflowrelevante Daten in einem Resource Repository gehalten. Somit müssen sie bei Verwendung in mehreren Prozessdefinitionen nicht neu definiert werden (Package Konzept). Alle Prozessdefinitionen in ein und demselben Package erben gemeinsame Attribute aus dem Prozesmodellbehälter, solange sie nicht „lokal überschrieben“ werden.

Externe Organisationsmodelle (OM), sprich Teilnehmer, Workflowrelevante Daten, Applikationen können aus einer XPDL Datei als „External Package“ importiert und in einer Prozessdefinition verwendet werden.

- **Deadlines / Ausnahmen**

Deadlines werden verwendet, um bei Überschreitung einer festgelegten Zeitperiode eine Ausnahme (engl. Exception) auszulösen. Im Falle einer Ausnahme wird eine weitere Aktivität zur Ausnahmebehandlung gestartet und der Workflow kann auf zwei Arten fortgesetzt werden: Entweder *asynchron* (die betroffene Aktivität wird fortgesetzt) oder *synchron* (die betroffene Aktivität bricht abrupt ab).

Bei der Modellierung eines Publikationsprozesses oder bei der Implementierung einer Spezifikation z. B. sind Deadlines ein wichtiges Charakteristikum.

- **Flexibilität und Erweiterbarkeit**

Es können verschiedene Ressourcetypen definiert werden und als Workflow Teilnehmer in einen Workflow eingebunden werden, unabhängig davon, ob Mensch oder Maschine. „Maschinen“ können darüberhinaus so genannte „generic tools“, oder „abstract tools“ sein. Dadurch ist es möglich Workflows auf unterschiedlichen Granularitätsebenen zu implementieren. So könnten z. B. folgende Funktionen als Anwendung in einem Workflow fungieren: `send_email`, `convert_document`, `analyse_document`.

- **Modellierung**

Für die Generierung von XPDL konformen Workflow Beschreibungen existieren einige frei verfügbare Editoren. Die bekanntesten sind wohl der *Java Workflow Editor (JaWE)*⁴ und das *XPDL definition module*⁵ für Microsoft Visio. Mit diesen Tools lassen sich XPDL konforme Dokumente graphisch erstellen und editieren.

Schwächen

Den Stärken von XPDL stehen folgende Schwächen gegenüber:

- **Verteilte Workflows**

In der Spezifikation XPDL existieren keine expliziten Konstrukte zur Beschreibung von verteilten Prozessabläufen.

6.2.2 WSFL, XLANG, BPEL4WS, WSCI

Da BPEL4WS eine Weiterentwicklung von WSFL und XLANG und WSCI vom Prinzip her kaum Unterschiede aufweist, werden diese Sprachen gemeinsam behandelt. Alle Sprachen sind auf die Komposition (Choreography & Orchestraion) von Web Services ausgerichtet.

Stärken

Die Stärken dieser Sprachen sind:

- **Trennung von Kontroll- und Datenfluss**

Bei Web Services ist es häufig der Fall, dass Kontroll- und Datenfluss divergieren. Falls der Datenfluss getrennt vom Kontrollfluss ist, muss in solchen

⁴<http://jawe.objectweb.org/>

⁵<http://www.capevisions.com/tech/wxml.shtml>

Situationen in WSDL keine generische Aktivität erzeugt werden, um die Daten zu transportieren.

- **Modellierung und Ausführung**

Mit BPWS4J⁶ von ALPHAWORKS existiert eine Laufzeitumgebung für BPEL4WS. Darin enthalten ist ebenfalls ein graphischer Editor als Plugin für die Eclipse⁷ Entwicklungsumgebung zum Erstellen und Bearbeiten von BPEL4WS Dokumenten.

Schwächen

An einem Workflow können immer nur durch WSDL beschriebene Ressourcen teilnehmen, wodurch menschliche Arbeitskraft kaum oder nur schwer einbezogen werden kann.

6.2.3 BPML

Die Bedeutung vom BPML hat in den letzten beiden Jahren stark nachgelassen. Immer mehr und größere Nutzer wechseln von BPML auf BPEL4WS. Im Gegensatz zu den anderen Sprachen hat BPML extensive Definitionsmöglichkeiten von Aktivitäten. Sie müssen nicht unbedingt atomar sein, sondern können rekursiv zu komplexen Aktivitäten ineinander gekapselt werden. In den anderen Sprachen ist dies nur auf Prozessebene mit Subprozessen möglich.

Obwohl die Sprache umfangreichere Definitionsmöglichkeiten als XPDL oder BPEL4WS hat, ist sie durch genauere Spezifikationen etwas unflexibler.

6.2.4 BPMN

Die Modellierungssprache BPMN ist noch sehr jung, erfreut sich aber immer mehr Beliebtheit. Durch so genannte Translatoren können BPMN Diagramme nach BPEL4WS, BPML und XPDL übersetzt werden und somit in einer für die jeweilige Sprache entworfenen Workflow Engine verwendet werden.

Zur Modellierung existieren bereits mehrere Plugins und Templates für Microsoft Visio. Zum einen das kommerzielle Produkt *Process Modeler for MS Visio*⁸ sowie frei erhältliche Templates⁹ für MS Visio von der weltweiten Workflow Community.

⁶<http://www.alphaworks.ibm.com/tech/bpws4j>

⁷<http://www.eclipse.org>

⁸<http://www.itpearls.com/en/commerce/products/modeler.shtml> [29.07.2004]

⁹<http://www.workflow-research.de/Downloads/BPMN/> [29.07.2004]

6.3 Fazit

Angesichts der Stärken und Schwächen vorhandener Beschreibungssprachen, ist XPDL die richtige Wahl für den Einsatz in VKC. Wichtiges Kriterium dafür ist die Möglichkeit, ein Organisationsmodell zu beschreiben, die Integration von Deadlines und die Einbindung beliebiger Ressourcetypen als Workflow Teilnehmer. Zusammen mit der frei verfügbaren Enhydra Shark Workflow Engine und dem JaWE Editor sind gute Voraussetzungen für eine erfolgreiche Implementierung und Integration in VKC gegeben. Zusammenfassend sind die ausschlaggebenden Punkte für diese Entscheidung:

- *XPDL* ist ein Standard der WfMC.
- Mit *JaWE* existiert ein guter graphischer Editor zur Erzeugung von XPDL-konformen Workflowdefinitionen.
- *Shark* stellt ein Workflow-Framework dar, arbeitet mit XPDL und unterstützt die Anforderungen an VKC. Zu dem ist die Entwicklergemeinschaft von Shark aktiv und reagiert sehr zügig auf Fragen, Wünsche und Benutzeranforderungen.

Alternativ bietet sich jBPM¹⁰ respektive der Beschreibungssprache jPdl und dem Editor jgpd an. jBPM orientiert sich stark am Referenzmodell und bietet ein hohes Maß an Flexibilität durch benutzerspezifische Erweiterungsmöglichkeiten. Falls man keinen großen Wert auf die Verwendung von Standards legt ist es eine echte Alternative zu XPDL/Shark/JaWE.

Für den Einsatz in heterogenen Web Services Umgebungen eignet sich dagegen am besten BPEL4WS. Durch ihre Vorgänger WSFL und XLANG hat BPEL4WS die Konzepte beider Sprachen übernommen und weiterentwickelt. Die Akzeptanz in der freien Wirtschaft ist groß.

¹⁰<http://www.jbpm.org>

7 Realisierung

In diesem Kapitel wird die Architektur der entwickelten Workflowkomponente skizziert und beschrieben, wie die gewonnenen Erkenntnisse aus der Evaluierungsphase in die Tat umgesetzt worden sind. Zunächst wird die Konzeption vorgestellt, dann werden die Methodiken bei der Implementierung beschrieben und zum Schluss das Ergebnis dargestellt.

7.1 Konzeption und Design

Die Konzeption und das Design der Workflowkomponente beruht auf der Wahl von XPDL als Workflow-Beschreibungssprache und Shark als Workflow Engine, welche sich am Referenzmodell der WfMC orientiert und eben XPDL als Beschreibungsformat verwendet.

7.1.1 Architektur

Bei dem Entwurf der Architektur stand zunächst die Frage im Raum, ob die Workflow Engine Shark als Bibliothek in VKC integriert werden oder stand-alone als Server fungieren sollte, mit welchem die VKC-Workflowkomponente dann via CORBA kommunizieren könnte. Die Entscheidung fiel auf Grund der nachfolgenden Gründe zugunsten der zweiten Variante:

- Bessere Skalierbarkeit der Gesamtanwendung, da die Workflow Engine auf einem anderen Rechner laufen kann
- Neben der VKC-Workflowkomponente können andere Klienten auf die Workflow Engine zugreifen
- Bei der Auslieferung von VKC *ohne* Workflowkomponente wird auch tatsächlich nur das Nötige ausgeliefert

Nachteilig wirkt sich dieser Ansatz aufgrund der Latenzzeit auf die Performanz aus, wenn VKC und Shark auf verschiedenen Rechnern ausgeführt werden. Insgesamt ergibt sich die in Abbildung 7.1 dargestellte Gesamtarchitektur.

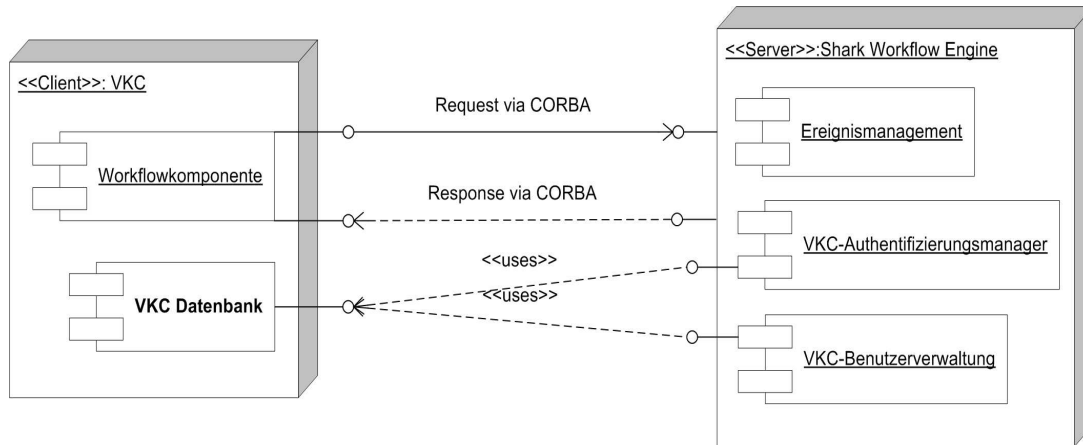


Abbildung 7.1: Gesamtarchitektur

Die Workflowkomponente wird als Modul in VKC integriert und kann Anfragen an die Workflow Engine via CORBA schicken oder auch Operationen wie „Workflow starten“ oder „Variable XY aktualisieren“ serverseitig ausführen.

Demgegenüber fungiert Shark als Workflow Server mit VKC-spezifischen Erweiterungen:

VKC-Authentifizierungsmanager stellt ein benutzerspezifisches Modul in Shark dar, welches für die Anmeldung von Benutzern an der Workflow Engine zuständig ist. Die Authentifizierung erfolgt dabei anhand der entsprechenden Daten in der VKC-Datenbank.

VKC-Benutzerverwaltung liefert eine Implementierung für die Benutzerverwaltung in Shark. Mit dem VKC-Authentifizierungsmanager und der VKC-Benutzerverwaltung kann Shark somit vollständig mit den VKC-Benutzern arbeiten.

Ereignismanagement ist eine Erweiterung der bestehenden Shark-Implementierung für die Persistenzierung von unterschiedlichen Ereignissen wie z. B. der Zuweisung einer Aufgabe an einen Benutzer. Damit ist es möglich bei Erhalt einer neuen Aufgabe eine Email an die entsprechenden Benutzer zu verschicken.

Die vorgesehenen Erweiterungen an Shark sind einfach zu realisieren, da Shark ein Workflow-Framework darstellt und diese Änderungen explizit vorsieht. Im Falle des VKC-Authentifizierungsmanagers und der VKC-Benutzerverwaltung muss lediglich eine von Shark bereitgestellte Schnittstelle implementiert werden. Der Sinn dieser Erweiterungen ist, Shark mit den bestehenden VKC-Benutzern arbeiten zu lassen.

7.1.2 Anwendungsfälle

Während der Designphase wurden zunächst mehrere Anwendungsfälle identifiziert und anschließend in Form eines *Use-Case-Diagramms* festgehalten. Das Resultat ist in Abbildung 7.2 zu sehen.

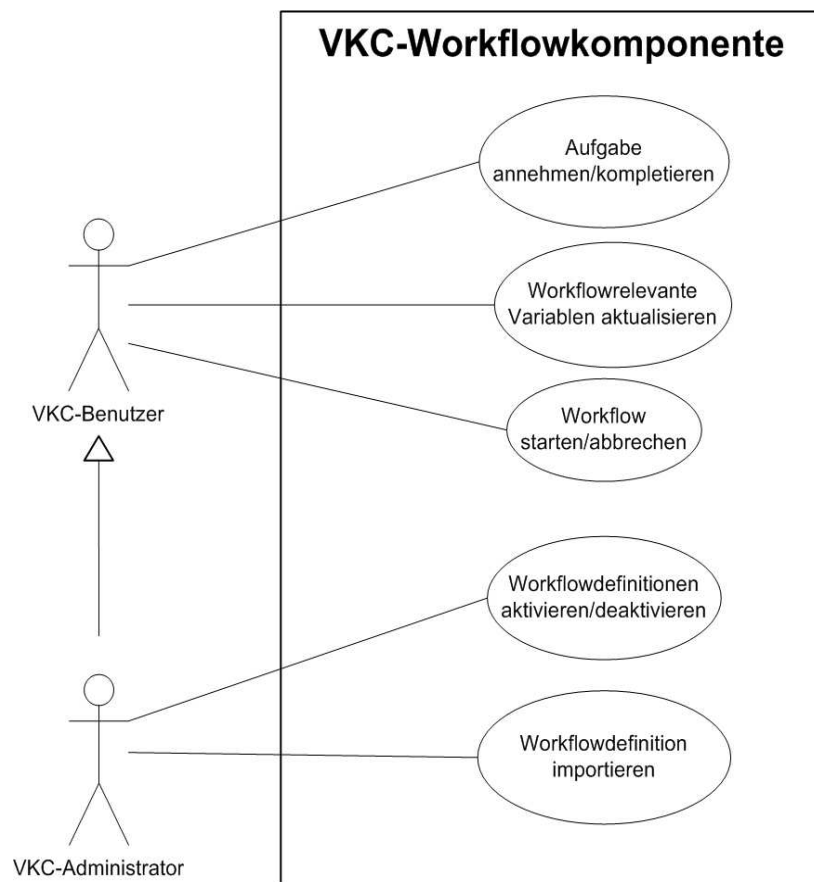


Abbildung 7.2: Anwendungsfälle der Workflowkomponente

- **Aufgabe annehmen/kompletieren**

Bei der Abarbeitung einer Prozessinstanz werden Aufgaben VKC-Benutzern oder -Gruppen zugewiesen. Diese werden dem Benutzer in verständlicher

Form mit allen verknüpften Daten angezeigt und es werden die Aktionen „Aufgabe annehmen“ und „Aufgabe erledigt“ zur Verfügung gestellt. In Anlehnung an das Referenzmodell der WfMC handelt es sich bei diesem Anwendungsfall um das Interface 2 (Worklist Handler, Abschnitt 4.4.4).

- **Variablen aktualisieren**

Ein sinnvoller Workflow enthält workflowrelevante Daten, die als Variable eines bestimmten Typs (Boolean, String, Integer, Date, etc.) repräsentiert werden. Eine Variable vom Typ *Date* wird z. B. verwendet, um eine Eskalationsbedingung zu überprüfen. Die Workflowkomponente stellt dem Benutzer die Möglichkeit zur Verfügung, die Werte solcher Variablen zu setzen oder in manchen Fällen nur einzusehen (z. B. eine Statusmeldung oder eine Anweisung zur Erledigung einer Aufgabe).

- **Workflow starten/abbrechen**

Zu den essentiellen Funktionen einer Workflowkomponente gehört neben dem *Worklist Handler* das Starten sowie Abbrechen von Workflows. Beim Starten eines neuen Workflows ist es möglich, VKC-Benutzer oder -Gruppen den in der Workflowdefinition enthaltenen Teilnehmer-Rollen zu zuordnen. Insgesamt geht es hier um das Interface 5 des Referenzmodells (Administration und Überwachung, Abschnitt 4.4.4).

- **Workflowdefinitionen aktivieren/deaktivieren**

Als Administrator hat man zusätzlich die Möglichkeit, einzelne Workflowdefinitionen zu aktivieren oder zu sperren. Damit können Workflowdefinitionen zwar in VKC geladen, aber nicht jedermann zur Instanziierung freigegeben sein.

- **Workflowdefinitionen importieren**

Zu guter Letzt ist für den Administrator eine Schnittstelle vorhanden, um in XPDL verfasste Workflowdefinitionen zu importieren (Interface 1, Prozessdefinition und -austausch, Abschnitt 4.4.4).

7.1.3 Systemanforderungen

Die Workflowkomponente wird als Modul in die bestehende VKC-Web-Anwendung integriert und sollte daher unter einem System mit folgender Konfiguration ausführbar sein:

- Windows 2000/XP, Linux oder Solaris

- JDK 1.4.2 oder höher
- Tomcat 4.x oder höher
- Firebird 1.5 oder höher

Da die Workflowkomponente lediglich eine Erweiterung von VKC ist, stellt die Einhaltung der Anforderungen kein Problem dar.

Die Shark Engine selbst läuft laut Angaben der Entwickler sowohl unter Windows als auch Linux. Die Richtigkeit dieser Angaben wird durch mehrere Benutzerbeiträge in der Shark-Mailingliste unter Beweis gestellt.

7.2 Implementierung

Nach der Designphase ist der Entwurf der Workflowkomponente sowie die vorgesehenen Erweiterungen der Shark Workflow Engine mit Hilfe von Standard-techniken implementiert worden. Dieser Abschnitt soll einen Überblick über die verwendeten Werkzeuge und Methoden geben.

7.2.1 Entwicklungsumgebung und Werkzeuge

Bei der Implementierung kam die Java-Entwicklungsumgebung Eclipse¹ mit einigen Plugins für die Entwicklung von Web-Anwendungen zum Einsatz. Darüberhinaus wurden die folgenden Werkzeuge in den Entwicklungsprozess mit eingebunden:

- **CVS**²

Mit diesem Versionsverwaltungswerkzeug wurde der Quellcode gepflegt. Nach Erreichen zuvor festgelegter Meilensteine wurden die Änderungen jeweils in das VKC CVS-Repository geschrieben.

- **Javadoc**

Der gesamte Quellcode wurde mit aussagekräftigen Kommentaren versehen, welche Grundlage für die mit *Javadoc* generierte API-Dokumentation sind. Diese findet sich auf der beiliegenden CD im Verzeichnis `/doc/api` wieder und ist für ein eventuelles Reengineering, für die Wartung und Weiterentwicklung der Workflowkomponente von großem Nutzen.

¹<http://www.eclipse.org/>

²<http://www.cvshome.org/>

- **Ant**³

Für den Build-Prozess ist das Build-Werkzeug *Ant* verwendet worden. Das bestehende Ant-Skript wurde um ein neues Target erweitert, welches die VKC-spezifischen Erweiterungen der Shark Engine kompiliert, zu einem Java-Archiv (JAR) packt und sie Shark zur Verfügung stellt – nach `SHARK_HOME/lib` kopiert.

- **Struts**⁴

Struts ist ein MVC⁵-Framework für die Entwicklung von Java Web-Anwendungen. Es ermöglicht die Trennung der Präsentationsschicht von der Geschäftslogik. Im nächsten Abschnitt wird genauer auf die Verwendung des Frameworks eingegangen.

- **Velocity**⁶

Velocity ist eine Template-Engine und wurde für die Generierung von Emails bei Zuweisung neuer Aufgaben verwendet.

- **Tomcat**⁷

Die Workflowkomponente wurde als Java Web-Anwendung realisiert und in dem Servlet-Container *Tomcat* ausgeführt.

7.2.2 Umsetzung

Geschäftslogik

Um die Implementierung flexibel zu gestalten wurde zu Beginn eine Schnittstelle für die Workflowkomponente festgelegt. Nach dem Factory-Designpattern können somit verschiedene Implementierungen realisiert werden. Dazu wurde die vorhandene VKC-Konfigurationsdatei um den Parameter `WorkflowServiceClass` erweitert, welcher als Wert den vollqualifizierten Klassennamen der jeweiligen Implementierung erwartet. Damit kann die Geschäftslogik der Workflowkomponente bequem ausgetauscht werden. Abbildung 7.3 zeigt den beschriebenen Sachverhalt.

Im Verlauf dieser Studienarbeit ist nur eine Implementierung umgesetzt worden, nämlich die „CORBA-Lösung“ (siehe Abschnitt 7.1).

³<http://jakarta.apache.org/ant/>

⁴<http://jakarta.apache.org/struts/>

⁵Model View Control

⁶<http://jakarta.apache.org/velocity/>

⁷<http://jakarta.apache.org/tomcat/>

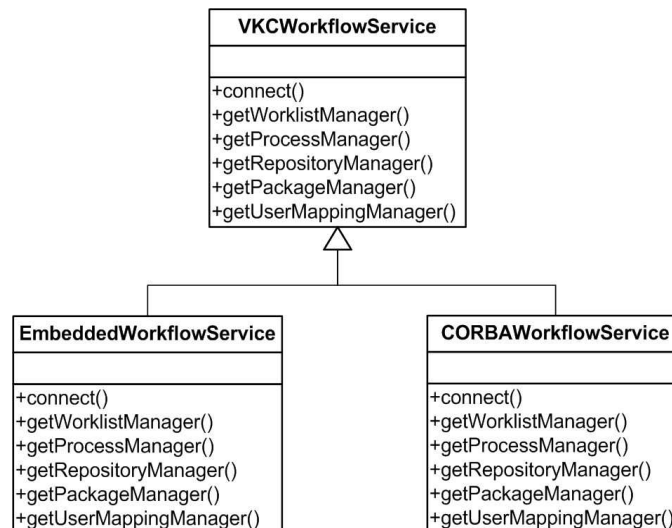


Abbildung 7.3: Wahl zwischen „eingebetteter“- und CORBA-Lösung

Benutzerschnittstelle

Neben der Geschäftslogik ist die Benutzerschnittstelle von entscheidender Bedeutung. Über sie kann der Benutzer in Interaktion mit der Workflow Engine treten und bestimmte Aktionen durchführen (z. B. einen neuen Workflow starten, eine Aufgabe als erledigt melden, etc.).

Um die Benutzerschnittstelle sauber von der Geschäftslogik zu trennen wurde das schon erwähnte MVC-Framework *Struts* verwendet. Die prinzipielle Arbeitsweise verdeutlicht Abbildung 7.4.

Als *Modell* dienen einfache *Java Beans*, die für die temporäre Datenhaltung zuständig sind und von der *Präsentationsschicht* dazu benutzt werden, eine Ausgabe für den Benutzer zu generieren. Die Aufgabe der *Kontrolle* übernehmen sogenannte *Action*-Klassen. In den meisten Fällen wird dort eine Verbindung zur Workflow Engine aufgebaut, eine Anfrage abgesetzt, das Ergebnis in Java Beans festgehalten und anschließend an die entsprechende JSP⁸ weitergeleitet.

7.2.3 Integration in VKC

Da die Workflowkomponente nicht zur Standardausrüstung von VKC gehören soll, wurde die bestehende VKC-Datenbank um zwei Systemparameter erweitert. Damit lässt sich zum einen die Workflowkomponente bequem über die Administrationsoberfläche an- und abschalten, zum anderen kann konfiguriert werden, ob

⁸Java Server Page

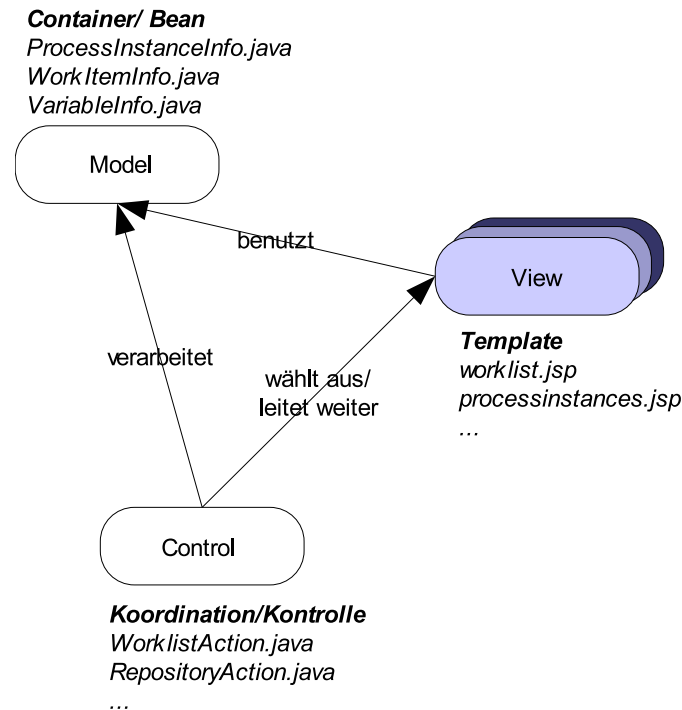


Abbildung 7.4: MVC Architektur

bei zugewiesenen Aufgaben eine Email an den/die beteiligten Benutzer verschickt werden soll.

Zusätzlich ist die VKC-Konfigurationsdatei um einige Parameter erweitert worden. Dort kann nun z. B. der Host und Port angegeben werden, unter dem die modifizierte Shark Engine läuft.

Die Workflowkomponente selbst wird (bei Aktivierung) in einer zusätzlichen Registerkarte in der VKC-Web-Anwendung eingeblendet (siehe Abbildung 7.6).

7.3 Praxisbeispiel

Die implementierte Workflowkomponente wurde anhand eines praxisnahen Beispiels getestet: Zum Abschluss eines jeden Jahres wird im C-LAB⁹ ein Jahresbericht angefertigt. Dieser Prozess umfasst dabei folgende Aktivitäten, die bislang manuell delegiert und überwacht werden.

- *Zuweisung* eines Verantwortlichen für ein Kapitel.

⁹<http://www.c-lab.de/>

- *Verfassung* des jeweiligen Kapitels, das damit verbundene *Korrekturlesen* und *Korrigieren*.
- *Erstellung* und Bearbeiten der Grafiken.
- *Übersetzung* ins Englische.
- *Zusammensetzung* des gesamten Jahresberichts zu einem PDF-Dokument.
- *Freigabe* zum Druck.

Eine Analyse des Prozessablaufs ergab zunächst den in Abbildung 7.5 skizzierten Teil-Workflow. Dieser ist nämlich bei der Erstellung eines jeden Kapitels gleich und wird folglich als *Subflow* im Gesamtworkflow zur Erstellung des Jahresberichts modelliert.

Der Gesamtworkflow besteht somit aus einer **Initialisierung**

- Bestimmung der Verfasser für die einzelnen Kapitel,
- Verknüpfung des Workflows mit einer VKC-Kategorie, welche als Projektverzeichnis für die hervorgehenden Dokumente dienen soll

und der **Freigabe zum Druck** sobald jeder Teil-Workflow – also die Fertigstellung eines einzelnen Kapitels – abgeschlossen ist.

Nach der Analyse des Geschäftsprozesses wurde der Workflow mit Hilfe von JaWE im XPDL-Format erstellt und anschließend getestet. Die endgültige Version befindet sich auf der beiliegenden CD im Verzeichnis `/praxisbeispiel`.

7.4 Test

System und Browser

Während der Entwicklung wurde sowohl die Workflowkomponente als auch die Shark Engine unter der folgenden Systemkonfiguration getestet:

- Windows XP SP1/SP2
- JDK 1.4.2
- Tomcat 4.1.29
- Opera 7.5

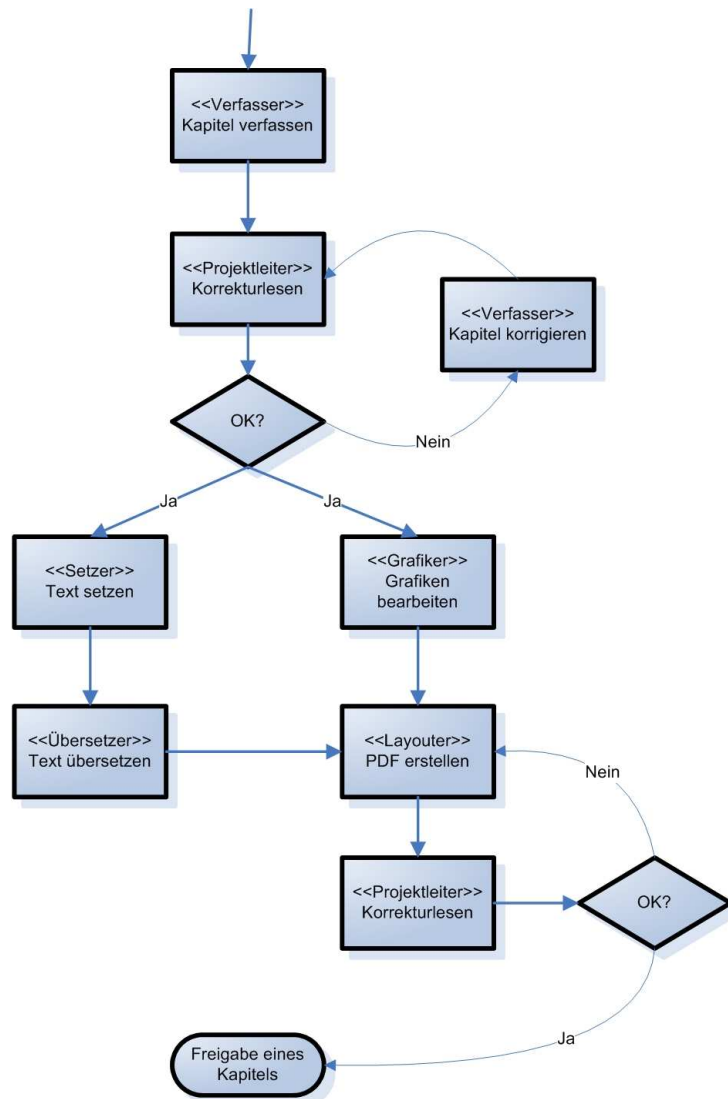


Abbildung 7.5: Vereinfachte Darstellung des Teil-Workflows zur Erstellung eines Kapitels beim Jahresbericht

Zusätzlich wurde die Benutzeroberfläche noch mit *Firefox 0.9* und dem *Internet Explorer 6* getestet. Darüberhinaus läuft VKC (mit integrierter Workflowkomponente) und Shark auch unter JDK 1.5.0 und Tomcat 5.x.x erfolgreich. Unter Linux oder anderen UNIX-Derivaten sind keine Tests gemacht worden.

Ressourcenverbrauch

In einem weiteren Test wurde der Ressourcenverbrauch überprüft. Lässt man sowohl VKC als auch Shark auf einem Rechner (1,5 GHz centrino, 512MB RAM) laufen, dann ist die Systemauslastung zum Teil so hoch, dass die Gesamtanwendung nicht performant genug ausgeführt werden kann. Nach einer gewissen Zeit

reicht der Arbeitsspeicher nicht mehr aus und Windows fängt mit der Auslagerung an. Verteilt man das System hingegen auf 2 Rechner eines lokalen Netzwerkes, dann werden die Lasten besser verteilt und die Antwortzeit bei Benutzerinteraktionen bleibt akzeptabel: 0,2 bis 0,8 Sekunden je nach ausgeführter Operation und angeforderter Datenmenge.

Konnektivität

Es wurde untersucht, wie sich die Workflowkomponente verhält, falls die Verbindung zur Workflow-Engine abreißt. In der aktuellen Fassung wird dabei noch eine Ausnahme ausgelöst, protokolliert und dem Anwender im Browser mit dem Hinweis, das Problem dem VKC-Administrator mitzuteilen, angezeigt. Letzteres ist nicht benutzerfreundlich und sollte durch eine verständliche Fehlermeldung ersetzt werden. Die Benachrichtigung des Administrators könnte automatisch per E-Mail erfolgen.

Steht die Verbindung zur Workflow-Engine wieder, so braucht die VKC-Anwendung nicht neu gestartet werden. Sie ist dem Benutzer sofort erneut in vollem Funktionsumfang zugänglich.

Benutzereingaben

Bei falschen oder ungültigen Benutzereingaben werden entsprechende Hinweise zurückgemeldet und dem Benutzer angezeigt. Ansonsten wird die Möglichkeit zur Eingabe falscher Daten vermieden, indem die Eingabe auf die Auswahl von gültigen Daten eingeschränkt wird. So wird zum Beispiel ein Wahrheitswert durch eine „Checkbox“ dargestellt oder die Auswahl einer VKC-Kategorie mit Hilfe eines Explorers unterstützt (siehe dazu Abschnitt 7.5, Abbildung 7.6 und 7.8).

7.5 Ergebnis

Aus der Implementierungsphase ist eine Workflowkomponente hervorgegangen, die in der vorläufigen Fassung aus genau vier Bereichen besteht:

1. einer persönlichen Arbeitsliste für jeden Benutzer
2. einem Administrations- und Überwachungsbereich für die initiierten Workflows eines Benutzers
3. einer Übersicht der vorhandenen Workflowdefinitionen, von der aus neue Workflows gestartet werden können

4. einem Repository für vorhandene Workflowdefinitionen sowie dazugehöriger Importfunktion von XPDL-Definitionen.

In den folgenden Kurzabschnitten wird über jeden Bereich ein Überblick in Form von Screenshots gegeben und die implementierten Funktionen erläutert.

Arbeitsliste

Die Arbeitsliste stellt dem Benutzer eine Übersicht der ihm zugewiesenen Aufgaben dar (siehe Abbildung 7.6).

The screenshot displays the 'Meine Arbeitsliste' (My Task List) interface. At the top, there is a navigation bar with tabs: 'Workflow', 'Favoriten', 'Historie', 'Neuerscheinungen', 'Bestseller', 'Erweiterte Suche', 'Messenger', and 'Prof'. Below this, the main content area is titled 'Meine Arbeitsliste'. It is divided into two main sections: 'Angenommene Aufgaben' (Accepted Tasks) and 'Ausstehende Aufgaben' (Outstanding Tasks).

Angenommene Aufgaben: This section contains a table with the following columns: 'Workflow', 'Aufgabe', 'Prio', 'Erhalten am', 'Dauer', and 'Aktionen'. One task is listed: 'Höhepunkte verfassen' (Workflow: Lesens/Freigabe, Prio: 3, Erhalten am: 22.09.2004, 23:56 Uhr, Dauer: 6 [min] 57 [s]). Below the table, there is a detailed view of the task, including a 'Beschreibung' (Description), 'Name', 'Kategorie' (Category), 'Status Initial', and 'Statusmeldung für Verfasser' (Status message for author). The 'Statusmeldung für Verfasser' field contains the text: 'Bitte noch einmal den 2. Abschnitt überarbeiten.' (Please revise the 2nd section once more).

Ausstehende Aufgaben: This section contains a table with the following columns: 'Workflow', 'Aufgabe', 'Prio', 'Ausstehend seit', 'Deadline', and 'Aktionen'. One task is listed: 'Vorwort verfassen' (Workflow: Manuelle Klassifizierung, Prio: 3, Ausstehend seit: 16.09.2004, 13:29 Uhr).

Several callout boxes with yellow backgrounds and red text are overlaid on the screenshot: 'Aufgabe erledigt!' (Task completed!) points to the 'Aktionen' column of the first task; 'Workflow visualisieren' (Visualize workflow) points to the 'Aktionen' column of the first task; 'Aufgabe weiterleiten' (Forward task) points to the 'Aktionen' column of the second task; and 'Aufgabe übernehmen' (Take over task) points to the 'Aktionen' column of the second task.

Abbildung 7.6: Screenshot der persönlichen Arbeitsliste

Dabei wird zwischen „angenommenen“ und „ausstehenden“ Aufgaben unterschieden. Eine Aufgabe gilt als „angenommen“ sobald sich der Benutzer entscheidet, eine ausstehende Aufgabe auch wirklich zu erledigen. Falls eine Aufgabe einer Gruppe zugewiesen wurde, so wird sie aus der Liste der „ausstehenden“ Aufgaben entfernt sobald ein anderer Benutzer aus der Gruppe die Aufgabe übernimmt. Dieser Mechanismus stellt sicher, dass eine Aufgabe nicht doppelt bearbeitet wird und ist in solchen Fällen nützlich, wo es egal ist, *wer* die Aufgabe erledigt, hauptsächlich *irgendjemand* erledigt sie (aber bitte nicht doppelt!). Darüberhinaus hat

der Benutzer die Möglichkeit, eine Aufgabe abzulehnen und sie an einen anderen Benutzer zu delegieren.

Zu jeder Aufgabe lassen sich diverse Einzelheiten und die assoziierten Variablen einblenden. Einige von ihnen werden nur angezeigt (read-only Variablen), andere hingegen können mit Abschluss einer Aufgabe aktualisiert werden und somit den Workflow beeinflussen.

Eine Besonderheit ist die Möglichkeit den umgebenden Workflow zu visualisieren. Dabei wird der Workflow als Grafik in einem neuen Fenster angezeigt und es werden die aktiven Elemente hervorgehoben. Somit behält der Benutzer den Überblick und weiß zu jedem Zeitpunkt, wie weit ein Workflow vorangeschritten ist (siehe Abbildung 7.7).

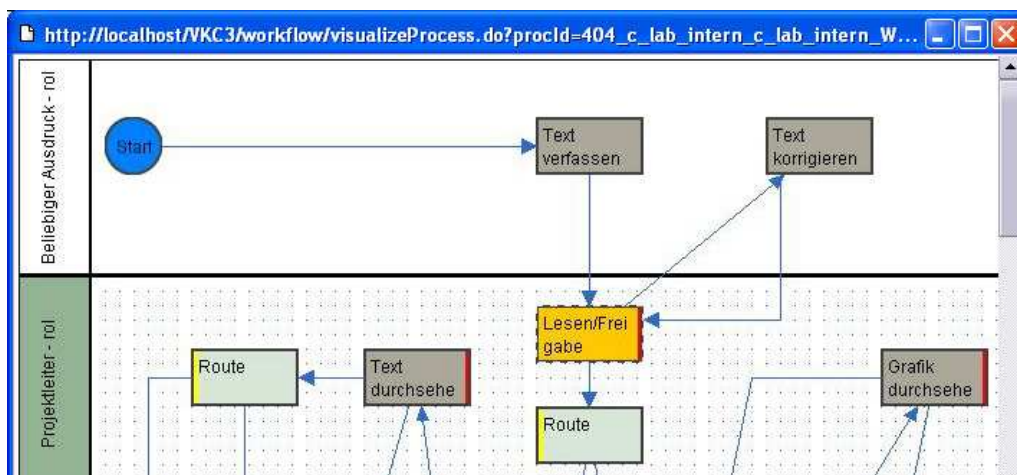


Abbildung 7.7: Visualisierung eines Workflows

Initiierte Workflows

Die Benutzeroberfläche für die Verwaltung der initiierten Workflows ist in Abbildung 7.8 dargestellt.

Der Benutzer hat in diesem Bereich die Möglichkeit,

- ...seine initiierten Workflows zu visualisieren.
- ...seine initiierten Workflows abubrechen.
- ...Workflowvariablen zu aktualisieren.

Bemerkenswert ist die Aktualisierung der verknüpften Kategorie: Variablen mit dem Namen oder der ID „category_id“ werden so dargestellt, dass der Benutzer die



Abbildung 7.8: Screenshot der Liste aktiver Workflows

gewünschte Kategorie mit Hilfe eines Explorers bequem und einfach selektieren kann. Wie in Bild 7.6 zu sehen ist, wird diese Variable in der Arbeitsliste dann als Verweis angezeigt.

Workflowdefinitionen

In einem weiteren Bereich der Workflowkomponente werden alle geladenen Workflowdefinitionen – in Paketen gruppiert – angezeigt. Neben den Standardaktionen

- Workflow starten
- Workflowdefinition graphisch darstellen
- Paket löschen (nur als Administrator möglich)
- Ganzes Paket oder Workflowdefinition (de)aktivieren (nur als Administrator möglich)

kann eine Zuordnung der definierten Rollen einer Workflowdefinition zu konkreten VKC-Benutzern oder -Gruppen getroffen werden (siehe Abbildung 7.9).

Zu beachten ist, dass die Zuordnung auf Paket- bzw. Definitionsebene stattfindet. Das bedeutet, dass die Zuordnung für *alle* Instanzen der jeweiligen Workflowdefinition gilt – auch für die schon laufenden. Möchte man für jede Instanz eine



Abbildung 7.9: Screenshot zur Zuordnung von Workflow Teilnehmern

eigene Zuordnung treffen, so muss man dazu eine Workflowvariable verwenden. Dieses Konzept wird im Anhang B.2 genauer erläutert.

Import von XPDL-Workflowpaketen

Neben den präsentierten Funktionen existiert noch die Möglichkeit in XPDL definierte Workflowbeschreibungen zu importieren, für die Benutzer freizugeben, zu sperren oder vollständig wieder zu entfernen.

8 Zusammenfassung und Ausblick

Im Verlauf der Studienarbeit ist eine Workflowkomponente entstanden, die vollständig auf dem prozessorientierten Ansatz (siehe Abschnitt 3.3) basiert und die von der WfMC propagierte Sprache XPD L als internes Definitionsformat verwendet. Dem voran wurde ausführlich die Architektur von Workflow Management Systemen beleuchtet und es wurden Workflow Beschreibungssprachen und freie Workflow Engines auf ihren Einsatz im VKC-Umfeld hin analysiert.

Eine Hürde stellten anfangs die Schlagwörter „Workflow“ und „Business Process“ dar. War bis vor wenigen Jahren noch nichts von „Business Process Management“ zu hören, ist es mittlerweile zu einem Modewort avanciert. Wo genau liegt der Unterschied zum Workflow Management? Gibt es überhaupt einen? Diesem Thema ist in Kapitel 4.1 ein kurzer Abschnitt gewidmet worden.

Zudem ist die Vielfalt an Workflow Beschreibungssprachen erdrückend. Jede betont verschiedene Aspekte eines Workflows und möchte zum Standard werden. Dies ist in naher Zukunft jedoch nicht absehbar. Nach wie vor hat man die Qual der Wahl und muss von Anwendungsfall zu Anwendungsfall unterscheiden, was mit Hilfe dieser Arbeit hoffentlich etwas erleichtert wird.

Obwohl die Workflow-Technologie seit mehr als 10 Jahren existiert, können damit bislang nur recht statische Workflows modelliert werden. Die wenigsten Workflow Engines können die Struktur eines Workflows zur Laufzeit ändern. Ein dynamisches Besprechungsprotokoll beispielsweise ist nicht zu realisieren. Nichtsdestotrotz hat die Technologie Zukunft und wird ein fester Bestandteil der meisten Softwareprodukte sein.

In der VKC-Umgebung ist in Zukunft eine weitergehende Integration der Workflowkomponente vorstellbar. Bei der Bearbeitung einer Aufgabe könnte der Benutzer dahingehend unterstützt werden, dass die Workflow Engine VKC-spezifische Funktionen via SOAP¹ aufruft, um bestimmte Aktionen auf Kategori-

¹Protokoll für die Ausführung von Operationen auf Web-Services

en oder Dokumenten auszuführen (z. B. eine Kategorie archivieren, ein Dokument konvertieren, den Status eines Dokumentes ändern, etc). Dies ist in der jetzigen Implementierung der Workflow Engine bereits vorgesehen, wurde zum Zeitpunkt der Implementierung wegen noch mangelnder Unterstützung seitens VKC aber noch nicht verwendet.

Desweiteren könnte der Zugriff auf VKC-Benutzer und -Gruppen von VKC und der Workflow Engine einheitlich über LDAP² erfolgen. Sobald VKC um eine LDAP-Schnittstelle erweitert wird, kann die Umstellung auf Grund der modularen Architektur von Shark sehr einfach vollzogen werden.

Abschließend sei erwähnt, dass die Arbeit Spaß gemacht hat und im C-LAB auf positive Resonanz gestoßen ist. Mehrere Mitarbeiter aus verschiedenen Abteilungen waren an den Zwischenergebnissen interessiert und wurden jeweils durch kurze Live-Demonstrationen informiert. Bei einer Kurz-Demo sprach der Verantwortliche für die Erstellung des Jahresberichts sogar vom „Haben wollen!“.

²Protokoll, um Benutzerinformationen von einem Server abzurufen

Abbildungsverzeichnis

3.1	Ruleflow Editor von ILOG JRules (Quelle: [JRu])	13
3.2	Prinzipielle Funktionsweise eines prozessbasierten Ansatzes	14
4.1	Beziehungen zwischen den grundlegenden Begriffen	20
4.2	Arten von Workflow (Quelle:[vdA98])	25
4.3	Merkmale eines Workflow Systems (Quelle:[Coa95])	28
4.4	Generische Struktur von Workflow Produkten (Quelle:[Coa95])	29
4.5	Das Workflow Referenzmodell (Quelle:[Coa95])	32
5.1	Metamodell von XPDL	37
5.2	Ein WSFL Flow Modell (Quelle:[Ley01])	41
5.3	Aktivitätstypen in BPML (Quelle:[Ark02])	47
5.4	Translation von BPMN Diagrammen	49
7.1	Gesamtarchitektur	59
7.2	Anwendungsfälle der Workflowkomponente	60
7.3	Wahl zwischen „eigebetteter“- und CORBA-Lösung	64
7.4	MVC Architektur	65
7.5	Vereinfachte Darstellung des Teil-Workflows zur Erstellung eines Kapitels beim Jahresbericht	67
7.6	Screenshot der persönlichen Arbeitsliste	69
7.7	Visualisierung eines Workflows	70
7.8	Screenshot der Liste aktiver Workflows	71
7.9	Screenshot zur Zuordnung von Workflow Teilnehmern	72

Anhang A Abkürzungen

ASAP - Asynchronous Service Access Protocol

AWSP - Asynchronous Web Services Protocol

BPEL4WS - Business Process Execution Language for Web Services

BPM - Business Process Management

BPML - Business Process Modeling Language

BPMN - Business Process Modeling Notation

CORBA - Common Object Request Broker Architecture

EAI - Enterprise Application Integration

JSP - Java Server Page

LDAP - Lightweight Directory Access Protocol

MVC - Model View Control

SOAP - Simple Object Access Protocol

UDDI - Universal Description, Discovery and Integration

WfMC - Workflow Management Coalition

WfMS - Workflow Management System

WSCI - Web Service Choreography Interface

WSDL - Web Services Description Language

WSFL - Web Services Flow Language

XML - eXtensible Markup Language

XPDL - XML Process Definition Language

Anhang B Erstellung einer XPDL- Prozessdefinition mit JaWE

In diesem Kapitel wird etwas detaillierter auf die Erstellung eines XPDL-Dokuments mit Hilfe von JaWE eingegangen. Im Vordergrund steht die Erläuterung einiger Besonderheiten von JaWE. Zur Vertiefung sei auf das Tutorial¹ von den JaWE Entwicklern verwiesen.

B.1 Paket-Attribute

Bei der Neu-Erstellung eines XPDL-Dokuments wird der Benutzer aufgefordert einige paket-spezifische Attribute zu setzen. Nicht bei allen Attributen ist der Zweck jedoch unmittelbar ersichtlich.

B.1.1 Publication Status

Mit diesem Attribut lässt sich der Status eines XPDL-Dokumentes überwachen. Die Prozessdefinitionen innerhalb eines Pakets können sich in den folgenden Phasen befinden:

- **UNDER_REVISION**: Revisionsphase
- **RELEASED**: Veröffentlicht
- **UNDER_TEST**: Testphase

¹<http://jawe.objectweb.org/doc/1.3/Tutorial/index.html> [9.08.2004]

B.1.2 Conformance Class

Die *Conformance Class* bzw. „Graph-Übereinstimmung“ beschränkt die Erstellung von Prozessdefinitionen auf bestimmte Klassen. JaWE warnt den Benutzer, falls bei der Erstellung die vorher eingestellte Konformität verletzt wird. Die Beschränkung gilt für alle Prozessdefinitionen eines Pakets.

- **FULL-BLOCKED**: Die Netzwerktopologie ist auf die korrekte Verschachtelung von SPLIT/JOIN und LOOP beschränkt.
- **LOOP-BLOCKED**: Die Netzwerktopologie ist auf die korrekte Verschachtelung von LOOP beschränkt.
- **NON-BLOCKED**: Es gibt keine Restriktionen bezüglich der Netzwerktopologie.

Wichtiger Hinweis: Falls Fehler bezüglich der Konformität vorhanden sind und sich das XPDL-Dokument *nicht* in der *Testphase* befindet, so lässt es sich nicht abspeichern.

B.2 Zuordnung von Workflow Teilnehmern

In XPDL definierte Rollen und Teilnehmer lassen sich mit der Shark Engine nur auf Ebene der Prozessdefinition auf die realen Benutzer des VKC-Systems abbilden. Dies bedeutet, dass *alle* Instanzen einer Prozessdefinition die gleiche Zuordnung von Rollen auf Benutzer/Gruppen haben. Möchte man den Rollen *unterschiedlicher* Instanzen *derselben* Prozessdefinition *verschiedene* Benutzer zuordnen können, so ist dies auf dem konventionellen Wege nicht möglich.

Lösung: Bei der Modellierung mit JaWE ordne man eine Aktivität der immer vorhandenen Rolle „Beliebiger Ausdruck“ zu. In den Eigenschaften der Aktivität kann man nun den Parameter „Ausführender“ auf den Namen einer zuvor definierten Workflow-Variable (Typ String) setzen. In Verbindung mit dem *VKCAssignmentManager* wird damit zur Laufzeit der Wert der Variablen als Benutzername betrachtet und die Aufgabe an genau diesen zugewiesen. Erlaubt ist zusätzlich eine durch Semikolons(!) getrennte Liste von Benutzernamen, also z. B. „kopyy;heinz;karl“. Gruppennamen werden in der jetzigen Implementierung nicht ausgewertet.

B.3 Aktualisierung und Ansicht von Variablen

Bei nahezu jedem sinnvollen Workflow müssen workflowrelevante Daten gesetzt, aktualisiert oder zumindest betrachtet werden können. Dies modelliert man in JaWE mit Hilfe erweiterter Attribute. Im Eigenschaften-Dialog einer Aktivität wechselt man zu den erweiterten Attributen und definiert die folgenden Attribute.

Aktualisierung: `VariableToProcess_UPDATE` und einen Variablennamen als Wert.

Ansicht: `VariableToProcess_VIEW` und einen Variablennamen als Wert.

Anhang C Konfiguration

C.1 VKC

Um die Workflowkomponente in VKC zu aktivieren, muss zunächst die vorhandene Konfigurationsdatei `VKC_HOME/defaultroot/WEB-INF/vkc.properties` um die folgenden Parameter erweitert werden:

```
vkc.workflow.enabled=true
vkc.workflow.host    =localhost
vkc.workflow.port    =10123
vkc.workflow.engine  =Shark
WorkflowServiceClass=vkc.workflow.service.CORBAWorkflowService
```

Bei dieser Beispielkonfiguration wird angenommen, dass die Shark Workflow Engine auf **localhost** unter Port **10123** läuft und sich durch den Namen **Shark** auszeichnet.

Zusätzlich muss die VKC-Datenbank mit Hilfe des SQL-Skripts `vkc_db_ib_add_workflow_ext.sql` um neue Systemparameter erweitert werden, die weitere Konfigurationsmöglichkeiten über die Administrationsoberfläche bieten. Damit wird ermöglicht, die Workflowkomponente zur Laufzeit bequem an- und abzuschalten oder die E-Mail-Benachrichtigung zu (de)aktivieren.

C.2 Shark Workflow Engine

Im folgenden sind die Parameter zu sehen, die erforderlich sind um Shark mit den VKC-spezifischen Erweiterungen zu starten.

```
# the name of shark instance - if shark is used in several VMs,
# this property MUST be different for each of them
enginename=Shark

# VKC Section
```

```
hibernate.connection.url      = jdbc:firebirdsql:localhost/3050:d:/vkc.gdb
hibernate.connection.driver_class = org.firebirdsql.jdbc.FBDriver
hibernate.connection.username  = sysdba
hibernate.connection.password  = xxxxxxxx
hibernate.connection.pool_size = 10

hibernate.C3PO_MAX_STATEMENTS = 0
hibernate.dialect              = cirrus.hibernate.sql.InterbaseDialect
hibernate.show_sql             = false
hibernate.statement_cache.size = 0
hibernate.use_outer_join       = true
hibernate.jdbc.use_scrollable_resultset = true

UserGroupManagerClassName=vkc.workflow.shark.VKCUserGroupManager
AssignmentManagerClassName=vkc.workflow.shark.VKCAssignmentManager
EventAuditManagerClassName=vkc.workflow.shark.VKCEventAuditManager
AuthenticationManagerClassName=vkc.workflow.shark.VKCAuthenticationManager
...
...
```

Besondere Aufmerksamkeit ist den letzten vier Parametern gegeben: Sie sind dafür verantwortlich, dass Shark mit den VKC-Implementierungen für die entsprechenden Manager gestartet wird. Fügt man oben gelistete Parameter also einfach in die bestehende Shark-Konfigurationsdatei `SHARK_HOME/conf/Shark.conf`, so ist darauf zu achten, dass die alte Konfiguration für den jeweiligen Manager auskommentiert wird. Ansonsten haben die Parameter `xxxManagerClassName` keine Wirkung.

Ferner müssen die VKC-Klassen natürlich im *Classpath* der JVM enthalten sein. Dies wird erreicht, indem die VKC-Klassen und andere notwendige Bibliotheken (z. B. Datenbank-Treiber) nach `SHARK_HOME/lib` kopiert werden. Eine Unterstützung hierfür liefert das Ant-Target `workflow.copy_vkclib`, wofür zuvor aber noch der Pfad zur Shark Installation gesetzt werden muss, also:

```
...
<property name="shark.home" value="d:/work/eclipse3/Shark"/>
...
```

Anhang D Inhalt der beiliegenden CD

Die beiliegende CD enthält...

- ...den kompletten Sourcecode der entwickelten Workflowkomponente
- ...die dazugehörige API-Dokumentation
- ...das Praxisbeispiel im XPDL-Format und als JPG-Grafik
- ...die vorliegende Studienarbeit im PDF-Format

Literaturverzeichnis

- [AE03] Stefan Fischer Andreas Eberhart. *Web Services: Grundlagen und praktische Umsetzung mit J2EE und .NET*. Carl Hanser Verlag, 2003.
- [AG03] ITpearls AG. Die neue welle des geschäftsprozess-managements. Whitepaper V1.0, ITpearls AG, <http://www.itpearls.com>, März 2003.
- [Ark02] Assaf Arkin. Business process modeling language. Technical report, BPMI.org, November 2002.
- [Bae04] Tom Baeyens. State of workflow. *TheServerSide*, Mai 2004. View at <http://www.theserverside.com/articles/article.tss?l=Workflow>.
- [BG00] Henriette Baumann and Patrick Grässle. *UML projektorientiert: Geschäftsmodellierung, Systemintegration, EDI*. Galileo Computing, 2000.
- [Boc93] G. Bock. Workflow as groupware. a case for group language? In *Groupware '93*, 1993.
- [BP84] G. Bracchi and B. Pernici. The design requirements of office systems. In *ACM Transactions on Office Information Systems*, volume 2, pages 151–170, 1984.
- [BP98] K. Barkaoui and L. Petrucci. Structural Analysis of Workflow Nets with Shared Resources. In W.M.P. van der Aalst, G. De Michelis, and C. A. Ellis, editors, *Workflow Management: Net-based Concepts, Models, Techniques and Tools (WFM'98)*, volume 98/7, pages 82–95. Eindhoven University of Technology, Eindhoven, 1998.

- [Bro94] J. Browne. A new approach to modularity in rule-based programming, 1994.
- [BS95] Markus Böhm and Wolfgang Schulze. Grundlagen von workflow-management-systemen. *Wissenschaftliche Beiträge zur Informatik, TU Dresden*, 2:50–65, 1995.
- [Coa95] Workflow Management Coalition. *The Workflow Reference Model*, Januar 1995. Document number TC00-1003.
- [Coa98] Workflow Management Coalition. *Workflow Management Application Programming Interface Specification*, version 2.0 edition, Juli 1998. Document Number WFMC-TC-1009.
- [Coa99a] Workflow Management Coalition. *Terminology & Glossary*, Februar 1999. Document Number WFMC-TC-1011.
- [Coa99b] Workflow Management Coalition. *Workflow Standard - Interoperability Abstract Specification*, November 1999. Document Number WFMC-TC-1012.
- [Coa01] Workflow Management Coalition. Workflow standard - interoperability wf-xml binding. Technical report, WfMC, November 2001. WFMC-TC-1023.
- [Coa02] Workflow Management Coalition. Workflow process definition interface - xml process definition language (xpdl). Technical report, Lighthouse Point, Florida, USA, 2002.
- [Coa04] Workflow Management Coalition. Introduction to the workflow management coalition. <http://www.wfmc.org/about.htm>, Juni 2004.
- [CW88] Thomas A. Cooper and Nancy Worgin. *Rule-based Programming with OPS5*. Morgan Kaufmann, San Mateo, California, 1988.
- [dbN04] dbNews. New business process modeling standard continues to gain momentum. *dbNews Denver*, Mai 2004. Gesichtet unter <http://www.dbbusinessnews.com/print.php?sid=2258> am 24.07.2004.

- [ea03] Satish Thatte et al. Business process execution language for web services specification. Specification 1.1, BEA Systems, IBM, Microsoft, SAP AG, Siebel Systems, Mai 2003.
- [EGR91] C.A. Ellis, S.J. Gibbs, and G.L. Rein. Groupware: Some issues and experiences. In *Communications of the ACM*, volume 43, 1991.
- [Fis02] Layna Fischer, editor. *Workflow Handbook 2002*. Workflow Management Coalition, 2002.
- [Fis04] Layna Fischer, editor. *Workflow Handbook 2004*. Workflow Management Coalition, 2004.
- [Gar03] Gartner. Gartner report, März 2003.
- [GRV99] Petra Gratzner, Christian Russ, and Peter Vaterl. Workflow standards. Universität Klagenfurt, 1999.
- [ITp03] ITpearls. Process modeler for ms visio. Technical report, ITpearls AG, März 2003.
- [JB96] S. Jablonski and C. Bussler. *Workflow Management. Modeling Concepts, Architecture and Implementation*. International Thomson Computer Press, 1996.
- [JBH97] S. Jablonski, M. Böhm, and W. Schulze (Hrsg.). *Workflow-Management: Entwicklung von Anwendungen und Systemen*. dpunkt.verlag, Heidelberg, 1997.
- [JRu] ILOG JRules. Business rule editors. <http://www.ilog.de/products/jrules/lifecycle/editor.cfm>.
- [KLR95] Gerti Kappel, Peter Lang, S. Rausch-Schott, and Werner Retschitzegger. Workflow management based on objects, rules, and roles. *Data Engineering Bulletin*, 18(1):11–18, 1995.
- [Kou95] Thomas M. Koulopoulos. *The Workflow Imperative*. Van Nostrand Reinhold, New York, 1995.
- [KR98] G. Kappel and W. Retschitzegger. The TriGS active object-oriented database system: An overview. *SIGMOD Record (ACM Special Interest Group on Management of Data)*, 27(3):36–??, ??? 1998.

- [Ley01] Prof. Dr. Frank Leymann. Web services flow language (wsfl 1.0). view or download <http://www-4.ibm.com/software/solutions/webservices/pdf/wsfl.pdf>, 2001.
- [Por02] Manfred Porak. Workflow. *Fachverband Unternehmensberatung und Informationstechnologie (UBIT)*, Juni 2002.
- [Ric03] Jeffrey Ricker. Asynchronous service access protocol. Working draft, OASIS, September 2003.
- [RT04] Daniel J. Hutzler Ralph Taugerbeck. Prozesse und portale. *Javamagazin*, 3:46–49, März 2004.
- [SBIS02] Sun, BEA, Intallio, and SAP. Web service choreography interface (wsci) 1.0. Technical report, W3C, August 2002.
- [Ser04] Siemens Business Services. Effizienter Informationsaustausch bei virtuellen Kooperationen durch Integration von Workflow- und Wissensmanagement-Techniken. Konzept im Rahmen einer Studie für die Universität Paderborn, Mai 2004.
- [SGP03] Keith D. Swenson, Mike D. Gilger, and Sameer Predhan. Wfxml 2.0 - xml based protocol for run-time integration of process engines. Working draft, WfMC, Oktober 2003.
- [Tec96] James Crawford Technologies. R++: Adding path-based rules to c++, 1996.
- [Tha01] Satish Thatte. Xlang- web services for business process design. Technical report, Microsoft Corporation, 2001.
- [uFS04] Kai Hermann und Frederik Stork. Regelbasierte Applikationen und Multiengine-Architekturen mit ilog jrules. *Javamagazin*, 2:78–82, 2004.
- [vdA98] Wil M.P. van der Aalst. The application of Petri nets to workflow management, 1998.
- [vdA03] Wil M. P. van der Aalst. Patterns and xpdL: A critical evaluation of the XML process definition language. QUT technical report, Queensland University of Technology, Brisbane, 2003.

-
- [vdABtHK00] Wil M.P. van der Aalst, A.P. Barros, A.H.M. ter Hofstede, and B. Kiepuszewski. Advanced workflow patterns. In O. Etzion and P. Scheuermann, editors, *7th International Conference on Cooperative Information Systems (CoopIS 2000)*, volume 1901, pages 18–29, Berlin, 2000. Springer-Verlag.
- [vdABV⁺99] Wil M. P. van der Aalst, Twan Basten, H. M. W. Verbeek, Peter A. C. Verkoulen, and Marc Voorhoeve. Adaptive workflow-on the interplay between flexibility and support. In *International Conference on Enterprise Information Systems*, pages 353–360, 1999.
- [wik04] Wikipedia, die freie enzyklopädie, Mai 2004. <http://de.wikipedia.org/>.
- [zMA00] Michael zur Muehlen and Rob Allen. Embedded vs. autonomous workflow - putting paradigms into perspective. In Layna Fischer, editor, *Excellence in Practice*, volume IV, pages 49–58. Lighthouse Point, 2000.