Aspektorientierte Komponentensysteme zur Unterstützung weitreichender Geschäftsprozesse

Zur Erlangung des akademischen Grades eines

Doktors

der Fakultät für Informatik

der Universität Karlsruhe (Technische Hochschule)

genehmigte

Dissertation

von

Rainer Schmidt

aus Erlangen

Tag der mündlichen Prüfung:	16.12.1999
Erster Gutachter:	Prof. Dr. Peter C. Lockemann
Zweiter Gutachter:	Prof. Dr. Wolffried Stucky

Kurzfassung

Der Prozeßgedanke hat besondere Bedeutung bei der Unterstützung der Zusammenarbeit von werden. Derartige, organisatorische Einheiten überschreitende Prozesse werden als mehreren Unternehmen oder Unternehmensteilen bekommen. Die dabei stattfindenden Abläufe können besonders gut als Prozesse dargestellt weitreichende (Geschäfts-)Prozesse bezeichnet werden. Weitreichende Geschäftsprozesse treten bei einer Vielzahl betriebswirtschaftlicher Organisationsformen auf wie beispielsweise in virtuellen Unternehmen oder Lieferketten.

Durch weitreichende Geschäftsprozesse ergeben sich neue Anforderungen an die informationstechnische Prozeßunterstützung, die von den bisherigen informationstechnischen Konzepten nicht erfüllt werden. Da die Teilnehmer eines weitreichenden Geschäftsprozesses autark bleiben wollen, ergibt sich die Anforderung, daß es in der Entscheidung der Prozeßteilnehmer liegen muß, mit welchen informationstechnischen Ressourcen sie die von ihnen bearbeiteten Teilprozesse unterstützen. Bei der Unterstützung weitreichender Prozesse ist mit tiefer greifenden Änderungen der Prozeßschemata zu rechnen als bisher. Die resultierenden Workflow-Schemata erfordern daher auch Erweiterungen der vom Workflow-Management-System unterstützten Menge von Schemaelementen, also des Workflow-Modells des Workflow-Management-System. Durch die häufigeren Prozeßänderungen ist die zudem bisher verwendete Totalabbildung von Geschäftsprozeß- auf Workflow-Schemata nicht mehr verwendbar. Die Unterstützung weitreichender Prozesse muß zudem umfassend sein. D.h., es muß eine derart leistungsfähige informationstechnische Prozeßunterstützung vorhanden sein, um alle Prozesse eines Unternehmens oder einer Verwaltung zu unterstützen.

In der Arbeit wird für die Bewältigung der obigen Problembereiche ein Lösungsansatz entwickelt, der aus vier Lösungskonzepten besteht: Dem Zwei-Ebenen-Workflow-Metamodell, dem Workflow-Modell-Wörterbuch, der Aspektelementorientierten Schemarepräsentation und einem inkrementellen Abbildungsverfahren für Geschäftsprozeßschemata.

Um die autarke Zuweisung informationstechnischer Ressourcen durch die Prozeßteilnehmer zu ermöglichen, wird ein Workflow-Metamodell mit zwei Ebenen geschaffen. Das Zwei-Ebenen-Workflow-Metamodell führt eine durchgängige Trennung zwischen einer spezifizierenden, logischen Ebene und einer implementierenden, physischen Ebene durch. Die logische Ebene legt die Grundlage für eine von konkreten informationstechnischen Ressourcen unabhängige Darstellung von Workflow-Schemata. Die physische Ebene schafft zu jedem Aspekt der logischen Ebene einen korrespondierenden Aspekt, der die informationstechnischen Ressourcen zur Unterstützung des logischen Aspekts darstellt. Durch die Verwendung von Aspektelementen als Granulat können für einzelne Elemente eines Aspektes unterschiedliche informationstechnische Ressourcen zur Unterstützung zugewiesen werden.

Zentrales Hilfsmittel zur dynamischen Verwaltung des Workflow-Modells ist das Workflow-Modell-Wörterbuch. Es ermöglicht es, zur Laufzeit die Menge der von einem WfMS unterstützten Schemaelemente zu erweitern. Der erste Schritt hierzu ist die Entwicklung eines Informationsmodells. Als Grundlage wird das im vorangegangenen Kapitel entwickelte Zwei-Ebenen-Workflow-Metamodell verwendet. Das Informationsmodell des Workflow-Modell-Wörterbuchs kennt 6 Objekttypen: Logische und physische Aspekte, logische und physische Elemente, Dienste und Ressourcen. Zwischen den logischen und physischen Aspekten und den logischen und physischen Aspektelementen bestehen Zugehörigkeitsbeziehungen. Zwischen je einem logischen und physischen Element besteht die Umsetzungsbeziehung, zwischen physischen Elementen und mehreren Diensten sowie mehreren Diensten und einem physischen Element die Unterstützungsbeziehung. Schließlich besteht zwischen Diensten und Ressourcen eine Bereit-

stellungsbeziehung. In einem weiteren Schritt wurden Operationen zum Anlegen der Objekt- und Beziehungstypen definiert.

Um die Anforderungen der Flexibilität und Skalierbarkeit zu erreichen, wird die aspekt-elementorientierte Schemarepräsentation entwickelt. Sie führt eine Zerlegung des Schemas in Elemente durch, die jeweils nur ein Aspektelement repräsentieren. Diese Elemente werden als Repräsentationselemente bezeichnet. Die Repräsentationselemente sind über sogenannte Verbinder miteinander verbunden. Es lassen sich zwei Arten von Verbindern unterscheiden. Die temporalen Verbinder geben zeitliche Abfolgen zwischen den Repräsentationselementen wieder. Nicht-temporale Verbinder geben Dienstnehmer-/Dienstgeberbeziehungen wieder. Die Verbindung zweier Repräsentationselemente über einen Verbinder setzt voraus, daß beide über dementsprechende Verbindungspunkte verfügen. Entsprechend der Einteilung bei den Verbindern wird zwischen temporalen und nicht-temporalen Verbindungspunkten unterschieden.

Die Herausforderung bei der Entwicklung des inkrementellen Abbildungsverfahrens liegt in der Bestimmung eines geeigneten Rasters für die Darstellung von Geschäftsprozeß- und Workflow-Modellen. Dazu werden drei Konzepte verwendet. Es sind dies die Aspektseparation, Klassifizierungsstrukturen und die Schaffung einer Äquivalenzsicht. Mit Hilfe der Aspektseparation kann ein erstes, grobes Raster geschaffen werden, das die Darstellung des Informationsgehalts von Geschäftsprozeß- und Workflow-Modell erlaubt. Durch Klassifizierungsstrukturen wird eine verfeinerte Darstellung der einzelnen Aspekte durchgeführt. Die Äquivalenzsicht legte die Grundlage für die Definition von Verknüpfungen zwischen semantisch äquivalenten, aber strukturell verschiedenen Konstrukten in Geschäftsprozeß- und Workflow-Modell. Auf der Basis des so geschaffenen Rasters konnten dann Verknüpfungen zwischen den Modellelementen definiert werden, die zur Definition von Abbildungsoperatoren führen.

Die auf diese Weise geschaffenen Abbildungsoperatoren erlauben eine inkrementelle Abbildung. Dazu wird das Geschäftsprozeßschema wird mit Hilfe der Äquivalenzsicht in ein semantisch äquivalentes Schema umgewandelt. Durch Anwendung der Abbildungsoperatoren wird dieses äquivalente Schema auf ein Workflow-Schema-Gerüst abgebildet. Schließlich wird das Workflow-Schema-Gerüst um Informationen ergänzt, die im Geschäftsprozeßschema fehlen, aber für ein vollständiges Workflow-Schema notwendig sind. Den Abschluß bildet eine Bewertung, in der die Durchführbarkeit der inkrementellen Abbildung mit den in diesem Kapitel geschaffenen Konzepten verifiziert wird.

Danksagung

Der größte Dank gilt meinen Eltern, die mir mein Studium ermöglicht haben und so die Grundlage für diese Arbeit legten.

Mein besonderer Dank auf fachlicher Ebene gilt Prof. Lockemann als Erstgutachter für die Betreuung der Arbeit. Trotz seiner umfangreichen Verpflichtungen setzte er sich intensiv mit meiner Arbeit auseinander. Prof. Stucky danke ich für die Übernahme des Korreferats und viele hilfreiche Anmerkungen. Bei beiden möchte ich mich für die stets angenehme Zusammenarbeit bedanken.

Erfolgreiche wissenschaftliche Arbeit kann nur in einem Umfeld gelingen, das Kritik und Anregung gleichermaßen bietet. Dieses Umfeld wurde mir von meinen Kollegen am FZI und IPD geboten, weswegen ihnen mein Dank gebührt. An erster Stelle ist Uwe Aßmann zu nennen, der seine Fähigkeiten als akademischer Lehrer eindrucksvoll unter Beweis stellte. Oliver Ciupke, Dirk Heuzeroth, Helmut Melcher, Rainer Neumann und Benedikt Schulz halfen mir durch kritische, aber zugleich konstruktive Diskussionen. Auch meinen Studenten Matthias David, Stefan Frei, Markus Hering, Jens Kaiser, Gunnar Scholz und Matthias Weichhold bin ich zu Dank verpflichtet. Stellvertretend für die anderen Kollegen in IPD und FZI möchte ich mich auch bei Claudia Rolker, Elvira Kuhn, Patricia Krakowski und Wassili Kazakos bedanken.

Nur schwer in Worte fassen läßt sich der Dank an Andrea für ihre Unterstützung und die immer neue Motivation in schwierigen Zeiten. Auch Maren möchte ich für Ihren Beistand danken.

Karlsruhe, im Januar 2000

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

Für immer mehr Unternehmen und Behörden wächst die Notwendigkeit, ihre Leistungserbringung in bezug auf Kosten, Zeit und Qualität zu optimieren. Durch die Globalisierung und die Öffnung von Märkten müssen sie gegenüber einer wachsenden Zahl von Mitbewerbern bestehen. Gleichzeitig wachsen die Transparenz und Dynamik der Märkte nicht zuletzt durch Medien wie das Internet. Daher wird von den Unternehmen eine immer schnellere Reaktion auf Aktionen der Mitbewerber gefordert.

Ähnliche Überlegungen gelten auch für Behörden, deren Effizienz und Geschwindigkeit wichtige Faktoren im Wettbewerb verschiedener Standorte darstellen. Verwaltungsakte werden zunehmend als Dienstleistungen begriffen, deren Qualität für den Bürger in der Verfahrensdauer und der Anzahl der notwendigen Behördengänge besteht. Daher sollen die Verwaltungsakte beschleunigt und die Zahl der Interaktionen des Bürgers mit der Verwaltung reduziert werden. Der Bürger soll nicht mehr eine Vielzahl von Behörden besuchen müssen, sondern den Verwaltungsvorgang nur anstoßen („one-stop-shopping“). Die Weiterleitung des Antrags und die Interaktion zwischen den Behörden soll dann von diesen selbst übernommen und nicht mehr dem Bürger aufgelastet werden. Der Bürger muß sich das dafür notwendige Wissen nicht mühsam aneignen.

Wichtigster Ansatzpunkt zur Optimierung der Leistungserbringung in Unternehmen und Behörden ist der Übergang von funktionalen Unternehmens- und Verwaltungsorganisationen zu Organisationen, bei denen Prozesse als zentrales Gestaltungsmittel dienen [HaCh95]. Das Ziel ist, die Leistungserbringung, beispielsweise die Herstellung eines Produktes [Loos96], die Erbringung einer Dienstleistung oder die Durchführung eines Verwaltungsakts [Sinz95b], in seiner Gesamtheit zu erfassen und zu optimieren. Im Rahmen der Prozeßorientierung werden die an der Leistungserbringung beteiligten Operationen nicht mehr isoliert betrachtet. Der Zusammenhang mit den anderen an der Leistungserbringung beteiligten Operationen wird optimiert. Dabei werden auch die zwischen ihnen bestehenden Kontroll- und Informationsflüsse zum Gegenstand von Verbesserungsmaßnahmen gemacht. Diese Betrachtung mehrerer Operationen und die Einbeziehung der Kontroll- und Informationsflüsse machen den wesentlichen Unterschied zur bisherigen, funktionalen Betrachtungsweise aus.

Allerdings reicht die einmalige Einführung der Prozeßorganisation nicht aus. So erfordert die Dynamik der Märkte die ständige Optimierung und damit Anpassung der Geschäftsprozesse. Behörden müssen geänderte Gesetze und Verwaltungsvorschriften umsetzen und dafür ebenfalls ihre Prozesse anpassen. Diese Veränderungen an Prozessen müssen schnell und effizient umgesetzt werden können. Auch wächst das Bewußtsein, daß das Wissen um die optimale Abfolge von Bearbeitungsschritten und Operationen ein wichtiges Kapital von Unternehmen und Behörden darstellt. Die Erfassung, Darstellung und Verbreitung dieses Wissens in Form von Prozessen kann daher als Art von Wissensmanagement verstanden werden.

Die Einführung von Prozessen als Gestaltungsmittel erlaubt deutliche Optimierungen dadurch, daß früher isoliert betrachtete Operationen in ihrem Zusammenhang als Bestandteile eines Prozesses gesehen werden. Es können Bezüge zwischen vormals isoliert gesehenen Informationen hergestellt werden, in dem sie als dem gleichen Prozeß zugehörig erkannt werden. Dazu ist es notwendig, Prozesse und ihre Ausprägungen als eigenständige Entität aufzufassen, die zustandsbehaftet ist und der ein Kontext zugeordnet ist. Der Prozeßzustand gibt an, wie weit die Bearbeitung einer Prozeßausprägung fortgeschritten ist. Der Prozeßkontext besteht aus der Prozeßausprägung zugeordneten Informationen, die jedoch eigenständige Entitäten darstellen. Ist der Zustand von Prozeßausprägungen erfragbar, so können hierdurch beispielsweise Kundenanfragen nach dem aktuellen Bearbeitungsstand eines Auftrags schnell beantwortet werden. Auch können durch Auswertung des aktuellen Prozeßzustands einer Teilmenge oder aller Prozeßausprägungen Handlungsempfehlungen abgeleitet werden. Ein Beispiel ist die Verstärkung des Personals einer Abteilung, die den Flaschenhals eines Prozesses bildet.

Während die Möglichkeit zur Abfrage des aktuellen Bearbeitungsstandes die Grundlage für ad-hoc-Maßnahmen wie beispielsweise die Zuordnung zusätzlicher Ressourcen schafft, ermöglicht die Aufzeichnung der Statusinformationen weitergehende Analysen. So kann der Prozeß auf wiederkehrende Schwachstellen hin untersucht werden. Der Prozeßkontext ermöglicht, daß bisher nur implizit vorhandene Bezüge zwischen Informationen die an der Ausführung eines Prozesses beteiligt sind, explizit dargestellt werden. Der Prozeßkontext besteht beispielsweise aus Auftragsinformationen, Kundenanfragen usw. Durch die explizite Darstellung des Prozeßkontexts wird eine bessere und umfassendere Informationsversorgung der Mitarbeiter ermöglicht, die im Kundenkontakt stehen. Sie können so schneller auf Kundenanfragen reagieren. Die Verlagerung des Prozeßkontexts ermöglicht es, Aufgaben schnell von einem Aufgabenträger auf einen anderen zu verlagern. Durch den Einsatz informationstechnischer Mittel kann dies geschehen, selbst wenn sich der andere Mitarbeiter an einem weit entfernten Standort befindet. Auf diese Weise kann in einem Unternehmen ein Lastausgleich zwischen verschiedenen Standorten und Abteilungen durchgeführt werden, der Kosten und Durchlaufzeiten zu reduzieren vermag.

Besondere Bedeutung hat der Prozeßgedanke bei der Unterstützung der Zusammenarbeit von mehreren Unternehmen- oder Unternehmensteilen bekommen. Die dabei stattfindenden Abläufe können besonders gut als Prozesse dargestellt werden. Derartige, organisatorische Einheiten überschreitende Prozesse sollen als weitreichende (Geschäfts-)Prozesse bezeichnet werden. Sie können sowohl innerhalb eines Unternehmens als auch zwischen Unternehmen bestehen. Der einfachste Fall ist dabei die Ausführung von Geschäftsprozessen über mehrere Niederlassungen hinweg. Eine lockere Form der Zusammenarbeit ist die Ausführung von Geschäftsprozessen über Tochtergesellschaften oder Profitcenter hinweg [RiNa97].

Die Zusammenarbeit von selbständigen Unternehmen ohne die Bildung von zusätzlichen Institutionen wird als virtuelles Unternehmen [Mert94] bezeichnet. Kennzeichnend für virtuelle Unternehmen ist ihre zeitliche Begrenztheit und ihre Ausrichtung auf die gemeinsame Erfüllung eines definierten Ziels, meist eines oder mehrerer Aufträge. Virtuelle Unternehmen erlauben die Bündelung der Kompetenzen der teilnehmenden Unternehmen und werden daher als besonders zukunftsweisende Form der Zusammenarbeit angesehen [Riem98]. Sie weisen eine tiefere Integration als strategische Partnerschaften auf, die meist nur Bereiche außerhalb der Kernkompetenz der teilnehmenden Betriebe in die Zusammenarbeit einbringen [MeFa95]. Über virtuelle Unternehmen hinaus gibt es noch lockere Formen der betrieblichen Zusammenarbeit, bei der weitreichende Geschäftsprozesse auftauchen, wie beispielsweise Kartelle oder Keiretsu [MeFa95].

Aus der informationstechnischen Perspektive wird häufig übersehen, daß auch die Zusammenarbeit zwischen Kunde und Anbieter einen weitreichenden Geschäftsprozeß darstellt.

Die Anbahnungsphase dieses Geschäftsprozesses wurde bis zum Aufkommen elektronischer Märkte nur konventionell, beispielsweise mit Briefen, Faxen, Telefonaten abgewickelt. Elektronische Märkte unterstützen die Anbahnungsphase, indem sie ein elektronisches Forum für die Plazierung von Angeboten bereitstellen. Zusätzlich ermöglichen sie dem Kunden, auf elektronischem Wege Aufträge zu erteilen. Auf diese Weise kann der, sonst an der Schnittstelle zwischen Kunde und Anbieter auftretende, Medienbruch zwischen papierbasierter und elektronischer Verarbeitung vermieden werden. Dieser Vorteil kann jedoch nur dann voll genutzt werden, wenn auch eine Anbindung des elektronischen Marktes an die Geschäftsprozesse des Unternehmens stattfindet.

1.1 Informationstechnische Prozeßunterstützung

Die Unterstützung von Prozessen und der mit ihnen verbundenen Konzepte, wie Prozeßstatus und Prozeßkontext, ist natürlich prinzipiell auch ohne Informationssysteme möglich. Durch deren Einsatz kann die Unterstützung jedoch viel effektiver geschehen. Zur informationstechnischen Unterstützung von Prozessen werden diese als Entität in den Entwurf und die Architektur von Informationssystemen eingebracht.

Idealerweise wird ein zweistufiger Ansatz angewendet, der aus einer fachlichen und einer informationstechnischen Sicht besteht [Ambe97]. Die fachliche Sicht wird im Rahmen eines Prozeßmodells erfaßt, die informationstechnische Sicht im Rahmen eines Workflow-Modells. Prozeß- und Workflow-Modell beschreiben die zur Bildung von Prozeß- und Workflow-Schemata zur Verfügung stehenden Modellelemente. Prozeß- und Workflow-Schemata stellen abstrahierende Beschreibungen eines Prozesses bzw. Workflows dar. Die Unterscheidung zwischen Prozeßmodell und Workflow-Modell ermöglicht eine klare Trennung von fachlichen und informationstechnischen Gesichtspunkten.

Prozeßmodelle für die Erfassung von Geschäftsprozessen werden Geschäftsprozeßmodelle [FeSi93] genannt. Prozeß- und Workflow-Modelle sind in Analogie zu Datenmodellen [LaLo95] zu sehen. Ein wesentlicher Unterschied von Prozeß- und Workflow-Modellen zu Datenmodellen ist, daß sich die Darstellung auf die auf Daten stattfindenden Operationen sowie deren zeitliche und ablauflogische Abfolge bezieht.

Auf der Grundlage von Prozeß- und Workflow-Modell werden Abbildungen konkreter Prozesse und Workflows in Form von Prozeßschemata und Workflow-Schemata gebildet. Prozeßschemata werden durch die Erfassung von Realprozessen geschaffen (siehe auch Abbildung 1). Sie stellen die fachliche Sicht des Prozesses dar. Die informationstechnische Sicht wird im Workflow-Schema wiedergegeben. Prozeß- und Workflow-Schema erhalten daher — nicht unbedingt deckungsgleiche — Informationen über den Realprozeß.

Workflow-Schemata geben die Prozeßschritte durch Workflow-Operationen wieder, die über Kontroll- und Datenflüsse miteinander verknüpft sind. Die Kontrollflüsse geben die zeit- und ablauflogischen Abhängigkeiten zwischen den Operationen wieder. Zeit und ablauflogische Abhängigkeiten drücken aus, wann welche Operation auf eine andere folgt, und unter welchen Bedingungen. Um ein Workflow-Schema zu erhalten, wird das Prozeßschema um Informationen ergänzt, die für die informationstechnische Unterstützung notwendig sind. Für die informationstechnische Unterstützung nicht benötigte Informationen werden entfernt.

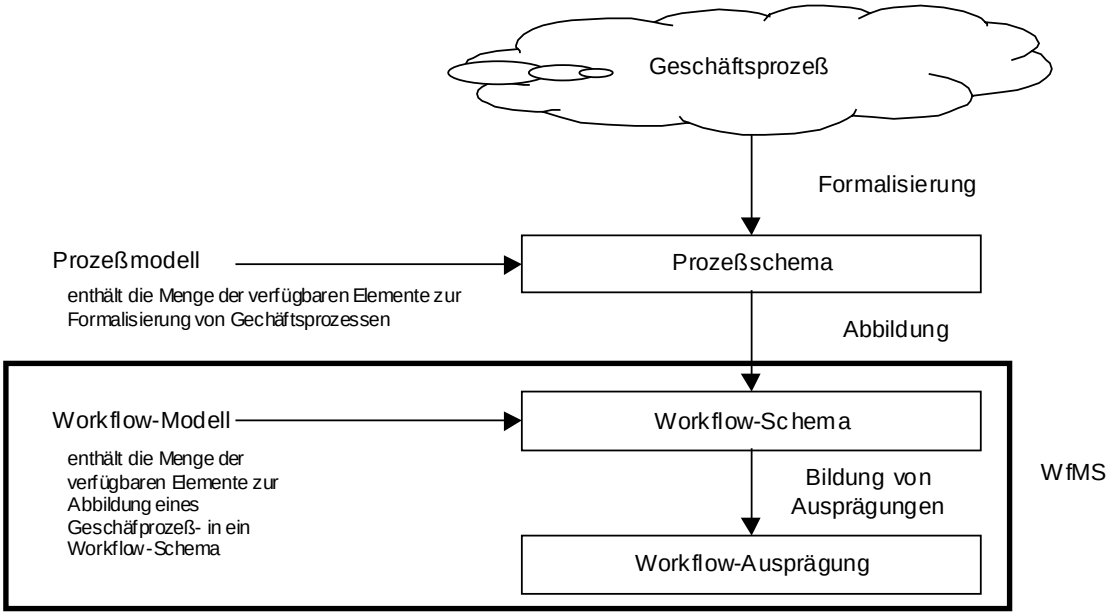


Abbildung 1: Informationstechnische Unterstützung von Prozessen

Die Aufgabe von Workflow-Management-Systemen (WfMS) ist es, Ausprägungen von Workflow-Schemata zu bilden und die gebildeten Workflow-Ausprägungen proaktiv auszuführen. Diesen Workflow-Schemata liegt ein Workflow-Modell zu Grunde, das die Menge der vom WfMS verarbeitbaren Schemaelemente definiert. Die proaktive Ausführung der Workflow-Schemata durch das WfMS bedeutet, daß nach Beendigung einer Workflow-Operation die nächste automatisch angestoßen wird und Daten zu ihr weitergeleitet werden. Bei der Ausführung von Workflows kann das WfMS meist externe Anwendungsprogramme, beispielsweise zur Durchführung einer Buchung, nutzen.

Der Entwicklung von WfMS konzentrierte sich bisher auf die flexible Umsetzung von Änderungen der Workflow-Schemata im Rahmen eines fest vorgegebenen Workflow-Modells. Änderungen der Geschäftsprozesse sollen schnell und mit wenig Aufwand umsetzbar sein. Stillschweigend wurden dabei mehrere implizite Annahmen gemacht, die bisher kaum näher betrachtet wurden, aber tiefgreifende Auswirkungen auf die entwickelten WfMS haben.

1. Die erste Annahme ist, daß alle Prozeßteilnehmer einer organisatorischen Einheit angehören und bereit sind, auf eine autarke Verwaltung der von ihnen bearbeiteten Prozesse sowie der zur informationstechnischen Umsetzung verwendeten Ressourcen zu verzichten.

2. Die zweite Annahme ist, daß Änderungen in den Prozeßschemata nur Änderungen von Workflow-Schemata, nicht aber des zugrundeliegenden Workflow-Modells erforderlich machen. Es wird davon ausgegangen, daß das Workflow-Modell a priori bestimmbar ist und keine späteren Erweiterungen benötigt werden.

3. Eine weitere Annahme bezüglich der Umsetzung von veränderten Prozeßschemata ist, daß diese Änderungen nur relativ selten erfolgen. Daher wird die Verwendung von gesamtabbildenden Verfahren, also Verfahren die nicht inkrementell arbeiten, für die Abbildung von Geschäftsprozeß- auf Workflow-Schemata für zulässig gehalten.

4. Die Zahl der Workflows ist durch die Beschränkung auf Abteilungsebene beschränkt. Eine umfassende Unterstützung, d.h. eine Unterstützung aller Prozesse, ist nicht notwendig.

1.2 Veränderte Rahmenbedingungen bei der Unterstützung weitreichender Prozesse

Bei weitreichenden Geschäftsprozessen gelten die im vorangegangenen Abschnitt identifizierten Annahmen jedoch nicht mehr. So kommt es zu veränderten Rahmenbedingungen für die informationstechnische Prozeßunterstützung, welche zu neuen Anforderungen an die Gestaltung der informationstechnischen Prozeßunterstützung führen.

1. Ein wichtiger Unterschied zur bisherigen Umgebung der Prozeß- und Workflow-Unterstützung ist die **Autarkie der Prozeßteilnehmer**. Aus ihr ergibt sich die Anforderung, daß es in der Entscheidung der Prozeßteilnehmer liegen muß, mit welchen informationstechnischen Ressourcen sie die von ihnen bearbeiteten Teilprozesse unterstützen. Insbesondere gehört dazu die Möglichkeit bereits vorhandene Ressourcen nutzen zu können. Es scheiden daher Konzepte aus, die den Prozeßteilnehmern die Art und Weise der informationstechnischen Umsetzung vorschreiben.

2. Bei der Unterstützung weitreichender Prozesse ist mit **tiefer greifenden Änderungen** der Prozeßschemata zu rechnen als bisher. Die resultierenden Workflow-Schemata erfordern dann auch Erweiterungen der vom WfMS unterstützten Menge von Schemaelementen, also des Workflow-Modells des WfMS. Gerade dies aber wird von den existierenden WfMS nicht unterstützt. So kommt es zu dem Paradox, daß ausgerechnet das Instrument zur Unterstützung des Wandels selbst wenig wandlungsfähig ist.

3. Eine weitere veränderte Rahmenbedingung ist, daß es zu **häufigeren Prozeßänderungen** als bisher kommt. Daher ist die bisher verwendete Totalabbildung von Geschäftsprozeß- auf Workflow-Schemata nicht mehr verwendbar. Bei ihrer Verwendung muß selbst bei kleinen Änderungen des Prozeßschemas dieses erneut in seiner Gesamtheit abgebildet werden. Ein solches Vorgehen ist bei kleinen und isolierten Schemata noch vertretbar. Bei umfangreichen und miteinander verknüpften Schemata ist es nicht mehr anwendbar, da eine Vielzahl sich fortpflanzender Änderungen erzeugt wird.

4. Die letzte veränderte Rahmenbedingung ist schließlich, daß nicht mehr nur eine Auswahl der Prozesse, sondern vielmehr ihre Gesamtheit unterstützt werden soll. Es muß also eine **umfassende** Unterstützung der Prozesse stattfinden. Daraus ergibt sich die Anforderung, daß eine derart leistungsfähige informationstechnische Prozeßunterstützung vorhanden sein muß, um alle Prozesse eines Unternehmens oder einer Verwaltung zu unterstützen. Diese Anforderung muß ohne Abstriche bei anderen Anforderungen, insbesondere der flexiblen Umsetzung von geänderten Prozeßschemata, erfüllt werden.

Es sind eine Reihe von Ansätzen entstanden, die eine oder mehrere der aus der Veränderung der Rahmenbedingungen resultierenden Anforderungen zu erfüllen versuchen.

1. Die Autarkie der Prozeßteilnehmer wird bisher nur unter Modellierungsgesichtspunkten untersucht, wie beispielsweise in [Riem98], [Ambe96b], [FeSi96a], [GrKa96]. Die Auswirkungen, die die Anforderung der Autarkie auf die Gestaltung von Software-Systemen zur Prozeßunterstützung besitzt, blieben bisher unberücksichtigt.

2. Das Problem der Erweiterbarkeit von WfMS wird vom Mobile-Ansatz [JaBS97] in Angriff genommen. Die erforderliche Erweiterbarkeit des Workflow-Modells zur

Laufzeit wird jedoch nicht erreicht, da eine Neuerstellung des WfMS zur Umsetzung von Erweiterungen notwendig ist. Der Mentor-lite [MWGW98] Ansatz versucht das Problem der Erweiterbarkeit durch die Repräsentation der Erweiterungen als Workflow zu lösen, verbaut dabei jedoch die Möglichkeit zur autarken Zuweisung informationstechnischer Ressourcen.

3. Das Problem der Abbildung von Geschäftsprozeß- auf Workflow-Schemata ist in einer Reihe von Arbeiten [Kreu96], [Lang96] angegangen worden. Bisher sind aber nur gesamtabbildende, d.h. nicht inkrementell arbeitende Abbildungen entwickelt worden.

4. Um eine umfassende Unterstützung von Prozessen zu ermöglichen ist die Skalierbarkeit der dazu verwendeten informationstechnischen Probleme sicherzustellen. Das Problem der Skalierbarkeit ist von einer großen Zahl von Arbeiten identifiziert worden [AlSc96], [CGPP97], [WWWK96], [SBMW96], [DKMS96], da die meisten WfMS zwar flexibel, nicht jedoch skalierbar sind. Daher versucht eine Vielzahl von Ansätzen die Skalierbarkeit zu verbessern. Herausragende Vertreter wie das Meteor- und Meteor2-Projekt [DKMS96] oder WorCOS-Ansatz [Schu97a], [Schu97c] erreichen nahezu eine skalierbare Workflow-Ausführung. Allerdings erreichen sie dieses Ziel nur um den Preis erheblicher Einschränkungen bei Flexibilität und Erweiterbarkeit.

Zusammenfassend läßt sich sagen, daß die Problematik der umfassenden Unterstützung weitreichender Prozesse in ihrer Gesamtheit nicht gelöst ist. Die existierenden Ansätze unterstützen daher das Ziel einer umfassenden Unterstützung weitreichender Prozesse nicht.

1.3 Hypothese der Arbeit

Hypothese der Arbeit ist, daß die Unterstützung weitreichender Geschäftsprozesse durch die Realisierung von drei Lösungskonzepten auf der Grundlage von Komponentensystemen erreicht werden kann. Durch die Lösungskonzepte wird als Granularitätskriterium für die Komponenten der Aspekt festgelegt, weswegen die sich ergebende Gesamtarchitektur als aspektorientiertes Komponentensystem bezeichnet wird. Ergänzt wird sie durch eine inkrementelle Abbildung von Geschäftsprozeß- auf Workflow-Schemata.

Bei den drei Lösungskonzepten handelt es sich um eine neuartige Gestaltung des Workflow-Metamodells, eine dynamisch erweiterbare Repräsentation des Workflow-Modells sowie eine zur Laufzeit anpaß- und erweiterbare Schemarepräsentation. Zur Realisierung dieser Lösungskonzepte ist eine Software-Architektur notwendig, die eine Reihe von nicht-funktionalen Anforderungen erfüllt, die sich aus den Lösungskonzepten ableiten. Diese Realisierungsgrundlage wird durch Komponentensysteme bereitgestellt.

1.4 Ziel der Arbeit

Ziel der Arbeit ist die Validierung der Hypothese durch die Schaffung einer Architektur zur Unterstützung weitreichender Geschäftsprozesse auf der Basis aspektorientierter Komponentensysteme. Diese wird durch ein inkrementelles Abbildungsverfahren für Geschäftsprozesse ergänzt.

Die zu schaffende Architektur soll nicht nur die Einbringung von Workflow-Schemaänderungen erlauben, die inkrementell aus einem geänderten Prozeßschema gewonnen worden sind, sondern auch dafür notwendige, zusätzliche Elemente des Workflow-Modells zur Laufzeit aufneh-

men können. Dies soll gleichzeitig mit einer hohen Skalierbarkeit des Ansatzes erreicht werden. Für alle Aspekte soll zudem die Autarkie bezüglich der informationstechnischen Umsetzung erreicht werden. Es soll auf feingranularer Ebene möglich sein, Ressourcen zur informationstechnischen Umsetzung der Prozesse einzusetzen.

1.5 Zu erwartende Beiträge der Arbeit

Die zu erwartenden Beiträge der Arbeit liegen in folgenden Punkten:

- Schaffung einer Architektur zur Unterstützung weitreichender Geschäftsprozesse. Wichtige Lösungskonzepte sind dabei:
    - Das Zwei-Ebenen-Workflow-Metamodell
    - Das Workflow-Modell-Wörterbuch[1]
    - Die aspektelementorientierte Schemarepräsentation
- Entwicklung eines Verfahrens für die inkrementelle Abbildung von Geschäftsprozeßschemata auf Workflow-Schemata
- Komposite Anwendungen als Bestandteil von Komponentensystemen
- Schaffung eines begrifflichen Instrumentariums für die Beurteilung von prozeßunterstützenden Software-technischen Systemen

Weitergehende Beiträge der Arbeit liegen in der terminologischen Erfassung von Komponenten und komponentenorientierten Rahmenwerken sowie Beiträgen zur allgemeinen Problematik des Entwurfs komponentenorientierter Anwendungen. Die hier vorgelegte Konzeption zur Unterstützung von Prozessen in Komponentensystemen kann zudem auch Anregungen für die Gestaltung anderer Anwendungen, wie beispielsweise elektronischer Märkte, auf der Basis von Komponentensystemen geben.

1.6 Struktur der Arbeit

Die Arbeit ist folgendermaßen auf Kapitel aufgeteilt. Die Argumentationslinie ist in Abbildung 2 dargestellt.

2. Grundlagen und Definitionen

Um die Grundlagen für die weitere Arbeit zu legen, werden zunächst Definitionen wichtiger Begriffe gegeben. Ausgangspunkt sind die Definitionen der Begriffe Modell, Schema und Ausprägung. Danach werden die Begriffe Prozeß, Geschäftsprozeß und Workflow definiert sowie voneinander abgegrenzt. Auch wird die Struktur von WfMS und ihre grundsätzliche Arbeitsweise verdeutlicht. Schließlich werden Bestandteile von Software-Systemen definiert und ihre Zusammenhänge geklärt.

3. Szenario

Zunächst werden in diesem Kapitel die in der Einleitung identifizierten besonderen Rahmenbedingungen für die Unterstützung weitreichender Prozesse mit einem Szenario verdeutlicht. Es handelt sich um die Auftragsbearbeitung in einem virtuellen Unternehmen. Auf der Basis dieser Darstellung werden dann informationstechnische Anforderungen für die Unterstützung weitrei-

[1] Der Begriff Wörterbuch steht für den englischen Begriff Repository

chender Prozesse abgeleitet. Diese bilden die Kriterien für die Beurteilung existierender Ansätze und sind zugleich Zielvorgabe für die in dieser Arbeit entwickelten Konzepte.

4. Stand der Forschung

Ziel dieses Kapitels ist es zu klären, welche Beiträge existierende Arbeiten liefern können. Kriterien sind die im vorangegangenen Kapitel identifizierten informationstechnischen Anforderungen aus der Unterstützung weitreichender Prozesse.

5. Lösungsansatz

Basierend auf der Untersuchung in Kapitel 4 wird eine Analyse durchgeführt, die aufzeigt, wieso die existierenden Ansätze nicht in der Lage sind, die gestellten Anforderungen in ihrer Gesamtheit zu erfüllen. Als Ergebnis wird ein Lösungsansatz mit 4 Bestandteilen entwickelt. Die ersten drei sind das Zwei-Ebenen-Workflow-Metamodell, das Workflow-Modell-Wörterbuch und die aspektelementorientierte Schemarepräsentation. Aus jedem dieser Lösungskonzepte ergeben sich nicht-funktionale Anforderungen, die durch das vierte Lösungskonzept, Komponentensysteme, erfüllt werden.

6. Zwei-Ebenen-Workflow-Metamodell

Zunächst wird das Zwei-Ebenen-Workflow-Metamodell entwickelt. Es legt die Grundlage für die autarke und flexible Zuweisung informationstechnischer Ressourcen bei der Workflow-Unterstützung. Aus dem Zwei-Ebenen-Workflow-Metamodell leitet sich die nicht-funktionale Anforderung der Ressourcenkonfigurierbarkeit her.

7. Workflow-Modell-Wörterbuch

Die dynamische Repräsentation des Workflow-Modells wird durch ein Workflow-Modell-Wörterbuch erreicht. Dieses ermöglicht die Erweiterung der unterstützen Workflow-Modellelemente zur Laufzeit. Zur Umsetzung des Workflow-Modell-Wörterbuchs müssen die nicht-funktionalen Anforderungen der unabhängigen Dienste- und Ressourcenerweiterbarkeit erfüllt werden.

8. Aspektelementorientierte Schemarepräsentation

Die aspektelementorientierte Schemarepräsention legt die Grundlage für die flexible Umsetzung von Workflow-Schemaänderungen bei gleichzeitiger Skalierbarkeit. Darüber hinaus können auch Erweiterungen um Schemaelemente vorgenommen werden, die durch Nutzung des Workflow-Modell-Wörterbuchs neu aufgenommen worden sind. Aus der aspektelementorientierten Schemarepräsentation erwachsen die nicht-funktionalen Anforderungen der Anwendungserweiterbarkeit und -konfigurierbarkeit sowie der Dienstekonfigurierbarkeit.

9. Komponentensysteme als Realisierungsgrundlage

Aufgabe dieses Kapitels ist es, mit Komponentensystemen eine Software-Architektur einzuführen, die die in den vorangegangenen drei Kapiteln identifizierten nicht-funktionalen Anforderungen erfüllt. Komponentensysteme bestehen aus kompositen Anwendungen, komponentenorientierten Rahmenwerken und Komponenten. Die zunächst eingeführten Konzepte der Komponente und des komponentenorientierten Rahmenwerks erfüllen die ermittelten nicht-funktionalen Anforderungen nicht vollständig. Daher wird mit kompositen Anwendungen eine Anwendungsarchitektur geschaffen, die die fehlenden Anforderungen, nämlich die unabhängige Anwendungserweiterbarkeit und –konfigurierbarkeit, erfüllt.

10. Aspektorientierte Komponentensysteme als Realisierung

Auf Grundlage der im vorangegangenen Kapitel entwickelten Komponentensysteme werden die Lösungskonzepte aus Kapitel 5 realisiert. Da dabei als Granularitätskriterium für die Komponenten des Komponentensystems Aspekte bzw. Aspektelemente dienen, wird der Begriff der aspektorientierten Komponentensysteme eingeführt. Zuerst wird die Realisierung der aspektelementorientierten Schemarepräsentation auf der Basis kompositer Anwendungen durchgeführt. In einem wieteren Schritt wird die Realisierung des Workflow-Modell-Wörterbuchs dargestellt.

11. Inkrementelle Abbildung von Geschäftsprozeßschemata auf Workflow-Schemata

In diesem Kapitel wird ein inkrementelles Abbildungsverfahren für Geschäftsprozeßschemata auf Workflow-Schemata entwickelt. Dazu wird zunächst ein geeignetes Raster für die Darstellung von Geschäftsprozeß- und Workflow-Schemata gebildet. Kernelemente sind die Aspektseparation von Geschäftsprozeß- und Workflow-Modell, die Feinzerlegung der Modelle durch Klassifizierungsstrukturen und die Schaffung einer Äquivalenzsicht. Mit Hilfe der Äquivalenzsicht können Verknüpfungen zwischen Geschäftsprozeß- und Workflow-Modell gebildet werden, die zur Bildung von Abbildungsoperatoren verwendet werden können.

12. Implementierung

In Kapitel 12 wird der Einsatz der in dieser Arbeit dargestellten Konzepte im Rahmen mehrerer Projekte dargestellt. Diese Projekte wurden größtenteils in Zusammenarbeit mit Industriepartnern durchgeführt. Daher können Erfahrungen aus dem operativen Einsatz der Konzepte wiedergegeben werden.

13. Evaluierung

Die Evaluierung geschieht durch den Nachweis, daß die in Kapitel 3 identifizierten entwickelten informationstechnischen Anforderungen für die Unterstützung weitreichender Prozesse von dem in dieser Arbeit entwickelten Konzept erfüllt werden. Dabei wird auch auf die Erfahrungen aus den in Kapitel 12 beschriebenen Projekten zurückgegriffen.

14. Zusammenfassung und Ausblick

In diesem Kapitel sollen die Ergebnisse der Arbeit zusammenfassend dargestellt werden und ein Ausblick auf weitere Arbeiten gegeben werden.

Schließlich soll noch auf den im Anhang befindlichen Index verwiesen werden, der unter anderem auf alle gegebenen Definitionen verweist.

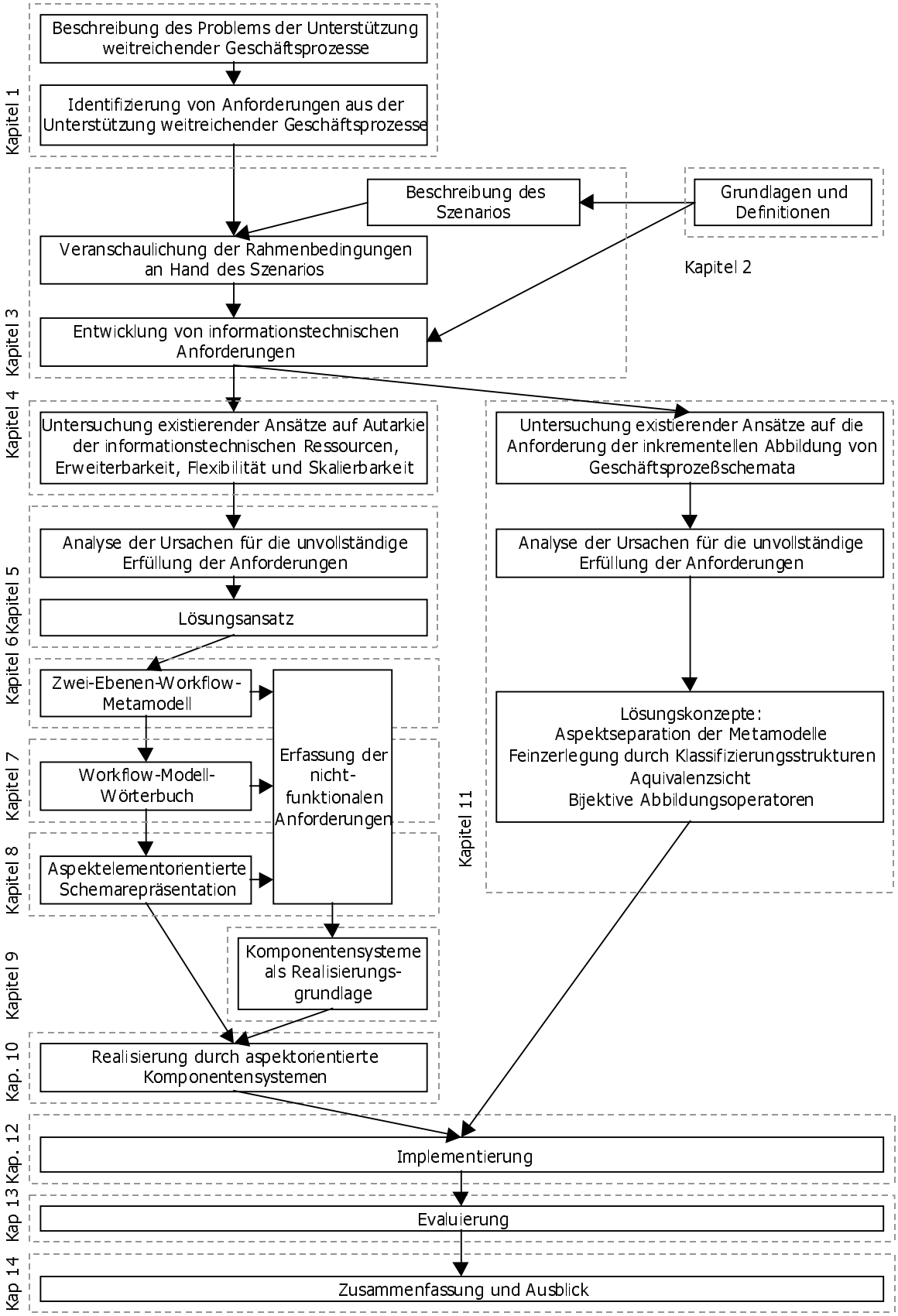


Abbildung 2: Argumentationslinie der Arbeit

2 Grundlagen und Definitionen

In diesem Kapitel sollen die terminologischen Grundlagen für die weiteren Kapitel dieser Arbeit gelegt werden. Dies geschieht in drei Bereichen.

Zunächst sollen die Begriffe Modell, Metamodell, Schema und Ausprägung definiert und zueinander in Beziehung gesetzt werden. Zu Grunde liegt das Verständnis, daß ein Modell ein System darstellt, welches ein anderes System, meist einen Ausschnitt aus der Realwelt, zielorientiert abbildet [BeSR95].

Aufbauend auf diesen Definitionen sollen dann Begriffe im Umfeld von Geschäftsprozessen und Workflows geschaffen werden. Ausgangspunkt ist die Trennung der fachlichen Sicht, vertreten durch Begriffe wie Prozeß oder Geschäftsprozeß, und der informationstechnischen Sicht, vertreten durch den Begriff des Workflows. Auf der Basis dieser Unterscheidung sollen zwei disjunkte Begriffssphären geschaffen werden, innerhalb derer weitere Begriffe eingeführt werden. Ziel ist es, (Geschäfts-)Prozesse von anderen Formen der Zusammenarbeit und Workflows von anderen informationstechnischen Konzepten abzugrenzen. Die Herausforderung liegt dabei darin, daß die Begriffe Prozeß, Geschäftsprozeß und Workflow häufig homonym und synonym verwendet werden. Homonyme Verwendungen des Begriffs Prozeß bezeichnen beispielsweise sowohl die fachliche Sicht auf Prozesse, als auch deren informationstechnische Umsetzung. Häufig synonym verwendet werden die Begriffe Geschäftsprozeß und Workflow wie beispielsweise in [Rein95], [Böhm97], [Loos96].

Der dritte Bereich umfaßt Begriffe, die der Beschreibung von Software-Systemen dienen. So gilt es Begriffe wie Dienst, Ressource usw. zu definieren und ihre Beziehungen untereinander zu klären.

2.1 Modell, Schema und Ausprägung

Da es bei grundlegenden Begriffen wie Modell, Schema, Ausprägung usw. zu homonymen Verwendungen in der Literatur kommt, sollen zunächst diese Begriffe geklärt werden. Dabei ist große Sorgfalt darauf zu verwenden, daß nicht Ausformungen des Modell-, Schema- oder Ausprägungsbegriffs in die Definitionen eingehen. Die Übertragung von Eigenschaften von Ausformungen des Modellbegriffs, beispielsweise in Form des objektorientierten und relationalen Modellbegriffs, ist daher zu vermeiden.

2.1.1 Modell

Kern des Modellbegriffes ist das Verständnis, daß es sich bei einem Modell um ein System handelt, welches ein anderes System zielorientiert abbildet [BeSR95]. Hierin unterscheidet sich

ein Modell von einem Artefakt, das kein anderes System abbildet, sondern aus sich selbst heraus existiert. Ein Modell ist eine Abstraktion, während ein Artefakt ein Konkretum ist. Wichtige Ziele der Modellbildung sind die Komplexitätsreduktion, Erklärung und Gestaltung des abgebildeten Systems [Fers94]. Die Zielorientierung eines Modells hat zur Folge, daß dem Modell als abbildendem System eine Perspektive zu Grunde liegt, die aus den mit der Modellbildung verfolgten Zielen entspringt. Diese Perspektive beeinflußt die Bildung der Abstraktionen aus denen sich das Modell zusammensetzt. Die gebildeten Abstraktionen werden als Modellelemente bezeichnet.

Definition 1 **Modell**
Ein Modell ist eine Menge von Abstraktionen, Modellelemente genannt, die ein System aus einer bestimmten Perspektive beschreiben.

Die Elemente eines Modells werden mit formalsprachlichen Mitteln wiedergegeben, sie können auch graphischer oder symbolischer Natur sein. Die Verwendung eines Modells und von Modellelementen wird in Abbildung 3 verdeutlicht. Das Modell M besteht aus den Modellelementen m_1, m_2 und m_3. Diese stellen Abstraktionen der Entitäten e_1'und e_1'', e_2'und e_2'' sowie e_3'und e_3'' des abzubildenden Systems dar.

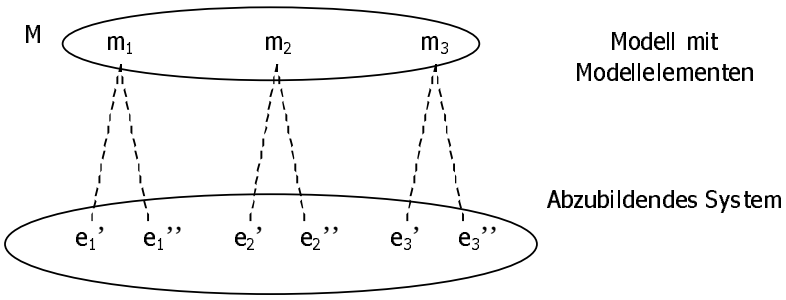


Abbildung 3: Zusammenhang zwischen Modell und abzubildendem System

Die Darstellung von Modellen geschieht vorteilhafterweise durch *Metamodelle*. Hierbei handelt es sich um ein Modell eines Modells. Ein Modell setzt sich daher aus Ausprägungen der Elemente des Metamodells zusammen. Ein Metamodell ist eine Menge von Abstraktionen, die zur Darstellung eines Modells verwendet werden können. So kann eine Modellierungssprache als Metamodell für die mit ihrer Hilfe gebildeten Modelle verstanden werden. Aus diesen Überlegungen ergibt sich folgende Definition des Begriffs Metamodell

Definition 2 **Metamodell**
Ein Metamodell ist ein Modell eines Modells.

Die Verwendung eines Metamodells wird in Abbildung 4 verdeutlicht. Das Metamodell MM besteht aus den Modellelementen mm_1, mm_2 und mm_3. Diese stellen Abstraktionen der Entitäten m_1'und m_1'', m_2'und m_2'' sowie m_3'und m_3'' des abzubildenden Modells dar.

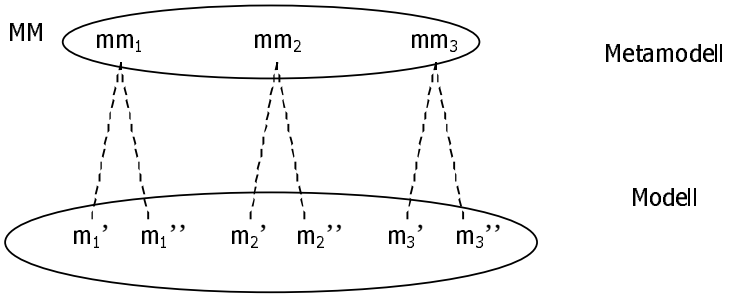


Abbildung 4: Zusammenhang zwischen Modell und Metamodell

Die Mißverständnisse aus der Verwendung des Begriffs Modell entspringen aus dem Umstand, daß der Begriff Modell für Darstellungen auf verschiedenen Abstraktionsebenen verwendet wird. Der Begriff Modell wird sowohl für die Menge zur Verfügung stehenden formalsprachlichen Elemente zur Darstellung eines Realweltausschnitts als auch für die Darstellung eines Realweltausschnitts selbst verwendet. Ein Beispiel für den ersten Fall ist das relationale Modell, daß von den Abbildungen der Realität, Schemata genannt, abstrahiert. Im zweiten Fall wird der Begriff des Modells auf Abbildungen der Realität verwendet. Ein Beispiel hierfür ist die Verwendung des Begriffs Geschäftsprozeßmodell in [FeSi93], bei der als Geschäftsprozeßmodell die Abbildung eines konkreten Geschäftsprozesses bezeichnet wird. Ähnliches gilt für Objektorientierte Modellierungssprachen, die das Ergebnis ihrer Anwendung als Modell bezeichnen. Diese Objektorientierten Modellierungssprachen sind daher eigentlich ein Metamodell.

2.1.2 Schema

Ein Schema ist ein Sonderfall eines Modells, das eine Menge von gleichartigen Entitäten in der Realwelt beschreibt. Modelle allgemein können dagegen auch die Darstellung von Abstraktionen zum Gegenstand haben kann, wie das Metamodell zeigt.

Definition 3 **Schema**
Ein Schema ist ein Modell, das eine Menge von gleichartigen Entitäten der Realwelt beschreibt.

2.1.3 Ausprägung

Auf der Basis eines Schemas können Ausprägungen gebildet werden. Eine Ausprägung[2] repräsentiert eine Entität in der Realwelt.

Definition 4 **Ausprägung**
Eine Ausprägung ist die Repräsentation einer Entität der Realwelt, die einem Schema entspricht.

Für diese Arbeit sollen die Begriffe Modell, Schema und Ausprägung folgendermaßen wie in Abbildung 5 veranschaulicht werden. Ein Modell stellt die formalsprachlichen Mittel zur Darstellung eines Systems aus einer bestimmten Perspektive heraus zur Verfügung. Mit diesen Mitteln werden Schemata gebildet, die ein Muster für einzelne Entitäten des abgebildeten Systems sind. Ausprägungen beschreiben genau eine Entität des abzubildenden Systems.

[2] Für den Begriff Ausprägung wurde bisher häufig der Begriff Instanz verwendet.

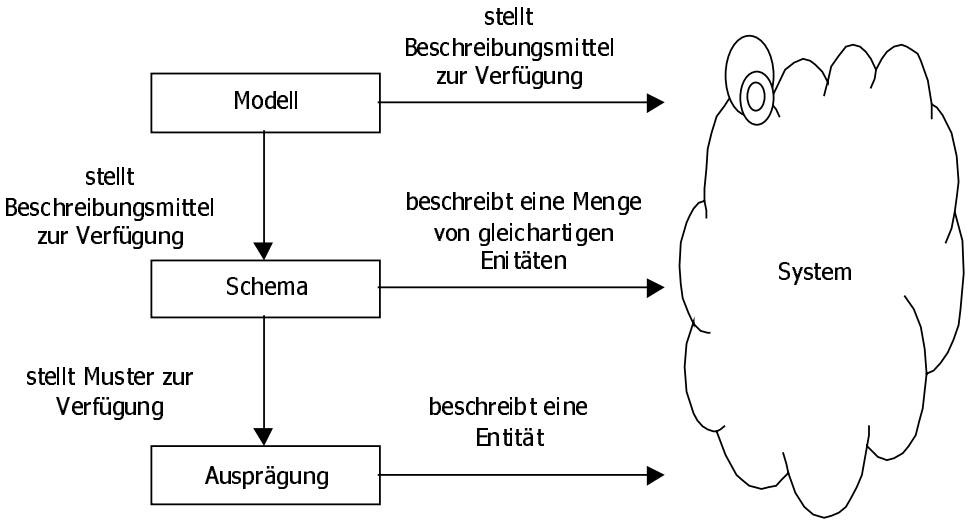


Abbildung 5: Modell, Schema und Ausprägung

2.2 Prozesse

Der Begriff Prozeß leitet sich nach [Dros89] vom lateinischen „processus“ her, also dem „Fortschreiten“ oder „Fortgang“. Bereits aus dieser etymologischen Betrachtung kann die grundlegende Bedeutung des Begriffes hergeleitet werden. Ein Prozeß besteht aus identifizierbaren, zeitlich und sachlich begrenzten Elementen, Schritten, die sich durch das Fortschreiten in einer Ordnung befinden. Die Schritte sind also identifizierbar. Die Halbordnung der Schritte besagt, daß sie auch parallel ausgeführt werden können. Für jeden Schritt läßt sich ein Vorgänger oder Nachfolger angeben, es sei denn es handelt sich um den ersten bzw. letzten Schritt.

Der Charakter des Begriffes Prozeß läßt sich weiter klären, indem man ihn mit den Begriffen Kommunikation, Kooperation und Koordination in Bezug setzt. Dazu müssen diese Begriffe jedoch selbst in ihrer Bedeutung geklärt werden. Kommunikation soll als Weitergabe von Informationen bezeichnet werden, ohne daß eine gemeinsame Aufgabe vorliegen muß oder gemeinsame Ressourcen genutzt werden. Auf Grundlage der Kommunikation finden Kooperation und Koordination statt, die sich als verschiedene Sichtweisen von Zusammenarbeit oder Interaktion auffassen lassen. Kooperation ist das gemeinschaftliche Lösen eines Problems auf der Basis einer gemeinsamen Menge von Daten oder Ressourcen. So wird in [LaLo95] Kooperation als das Zusammenwirken über gemeinsame Datenbestände hinweg definiert. Kooperation betrachtet also in erster Linie die Ressourcensicht.

Bei der Koordination stehen nicht Daten oder Ressourcen, sondern die zum Erreichen eines Ziels ausgeführten Operationen oder Aktivitäten im Mittelpunkt [Schä96]. So wird Koordination in [Gail95] als die geeignete Abstimmung von Operationen auf Daten bezeichnet. Allgemeiner ist die Definition in [HaSy95], die Koordination als „die Abstimmung von Einzelaktivitäten im Hinblick auf ein Ziel“ definiert. In [MaCr94] wird Koordination als die Verwaltung von Abhängigkeiten zwischen Aktivitäten erklärt. Es kann zwischen technischer Koordination, also beispielsweise Sperrmechanismen in Datenbanken, und semantischer Koordination, die sich aus fachlichen Anforderungen ergibt, unterschieden werden. Fußend auf diesen Definitionen wird deutlich, daß Prozesse mehr als nur kommunizierende Einzelaktivitäten sind, nämlich eine Interaktion mit kooperativen als auch koordinativen Elementen. Kooperativ ist das Lösen einer Aufgabe auf der Basis einer gemeinsamen Menge von Daten, koordinativ ist die Abstimmung der einzelnen Schritte zum Lösen der Aufgabe.

Prozesse können gegenüber anderen Formen des Zusammenwirkens mit Hilfe einer Raum/Zeit-Taxonomie [Joha91] abgegrenzt werden (siehe Tabelle 1). Diese Taxonomie unterscheidet Formen des Zusammenwirkens danach, ob sie am gleichen Ort oder verschiedenen Orten, und ob sie gleichzeitig oder zeitlich versetzt erfolgen. Wenn das Zusammenwirken am gleichen Ort und zur gleichen Zeit stattfindet, spricht man vom lokalen Zusammenwirken. Beispiele sind Konferenzen und Besprechungen. Sie bedürfen keiner direkten informationstechnischen Unterstützung. Dies gilt auch für das lokale, aber zeitlich versetzte Zusammenwirken, nur mit dem Unterschied, daß Hilfsmittel, wie beispielsweise ein Anschlagbrett, benötigt werden. Das Zusammenwirken an verschiedenen Orten benötigt jedoch Kommunikationsmittel, die vom Telefon bis zum Videokonferenzsystem reichen können. Die vierte Form des Zusammenwirkens, nämlich das zeitlich versetzte und verteilte Zusammenwirken, ist typisch für Prozesse. Sie stellt die höchsten Anforderungen an die Unterstützung, da sowohl örtliche als auch zeitliche Unterschiede überbrückt werden müssen.

	Gleichzeitig	Zeitlich versetzt
Am selben Ort	Lokales Zusammenwirken Beispiel: Konferenz	Asynchrones Zusammenwirken Beispiel: Anschlagbrett
An verschiedenen Orten	Synchrones verteiltes Zusammenwirken Beispiel: Telefon	Asynchrones verteiltes Zusammenwirken: Prozeß

Tabelle 1: Formen des Zusammenwirkens von Personen

Ein weiteres wichtiges Element von Prozessen ist die Betrachtung der Prozeßschritte unter zwei Perspektiven: Erstens als Definition von Aufgaben, und zweitens als Zuweisung von Aufgabenträgern, die diese Aufgaben erfüllen. Hierdurch wird eine Indirektion möglich, die eine dynamische Zuordnung von Aufgaben und Aufgabenträgern ermöglicht. Die Ausführung einer Aufgabe durch einen Aufgabenträger soll als *Aktivität* bezeichnet werden.

Auf Basis dieser Begriffsklärungen kann die Definition des Begriffs Prozeß gegeben werden:

Definition 5 **Prozeß**
Ein Prozeß besteht aus einer Menge von Aktivitäten, die zur Erfüllung eines Zieles zusammenwirken. Dieses Zusammenwirken findet asynchron und verteilt statt und enthält sowohl kooperative als auch koordinative Elemente. Kooperativ an einem Prozeß ist die Verfolgung eines Ziels auf der Basis einer gemeinsamen Menge von Informationen oder Ressourcen. Die dazu notwendigen Aktivitäten werden zur Erreichung des Ziels koordiniert. Durch die Koordination und Kooperation wird eine Halbordnung auf den Aktivitäten des Prozesses definiert.

Ausformungen von Prozessen finden sich nicht nur im betriebswirtschaftlichen Bereich in der Form von Geschäftsprozessen, sondern auch im technischen Bereich in Form von Entwicklungsprozessen, wie beispielsweise in der Software-Entwicklung.

Das Wissen um die optimale Gestaltung von Prozessen stellt eine häufig übersehene — und in ihrer Bedeutung verkannte — Form von Wissen dar. So ist nicht nur das Wissen um die korrekte Ausführung von einzelnen Prozeßschritten, sondern auch das Wissen über die zeit- und sachlogischen Abhängigkeiten zwischen den Prozeßschritten für den erfolgreichen Prozeßabschluß entscheidend. Eine geeignete Darstellung dieses Wissens ist daher notwendig, um es explizit zu machen und weitergegeben zu können.

2.2.1 Geschäftsprozesse

Die Unklarheiten über den Begriff Geschäftsprozeß rühren daher, daß eine Vielzahl von Definitionen existiert, die zudem aus unterschiedlichen Perspektiven stammen. Eine dieser Perspektiven ist stark auf die Optimierung der betrieblichen Wertschöpfung ausgerichtet. So wird in [HaCh95] ein Geschäftsprozeß als „Bündel von Aktivitäten, für das ein, oder mehrere unterschiedliche Inputs benötigt werden und das für den Kunden ein Ergebnis von Wert erzeugt“ gesehen. Im Zentrum des Interesses steht also die Schaffung eines Nutzens für den Kunden, wogegen die Beschreibung der Bestandteile des Geschäftsprozesses und seiner Struktur vage bleibt.

Der Begriff Geschäftsprozeß wird in diesem Umfeld daher auch auf Abläufe angewendet, die nach der obigen Definition gar keine Prozesse sind. Ist dies in der betriebswirtschaftlichen Literatur noch zu erwarten, so überrascht es, sie auch in der Informatikliteratur anzutreffen. Dabei entstehen Brüche in der Terminologie, denn der Begriff Prozeß wird meist ähnlich wie in Definition 5 verwendet. So wird in [JaBS97] ein Prozeß als ein „..Vorgang zu dem es eine Beschreibung (Schema) gibt.“ definiert. Geschäftsprozesse hingegen werden nicht, wie man es erwarten würde, als Prozeß in einer Wirtschaftseinheit definiert, sondern als bloßer Vorgang: „ ein Geschäftsprozeß ist ein Vorgang in Wirtschaftseinheiten ...“. Geschäftsprozesse müssen also nach dieser Definition kein Prozeß sein. Die folgende Definition bezieht sich dagegen schon stärker auf die koordinative Natur eines Prozesses: „Unter einem Prozeß wird die zeitlich-sachlogische Abfolge von Funktionen, die zur Bearbeitung eines betriebswirtschaftlich relevanten Objekts notwendig sind, verstanden” [BeVo96]. Eine detailliertere Sicht findet sich in der Definition von [Dave93] ,, .. a process is simply a structured, measured set of activities designed to produce a specific output for a particular customer or market. .. .. A process is thus a specific ordering of work activities across time and space, with a beginning, an end, and clearly identified inputs and outputs: a structure for action“. Diese Definition sieht den Geschäftsprozeß nicht nur als Bestandteil der betrieblichen Wertschöpfung, sondern auch als Anordnung zeitlich begrenzter, betrieblicher Aktivitäten. Diese stehen in zeit- und sachlogischen Abhängigkeiten und es findet ein geregelter Austausch von Informationen zwischen ihnen statt. Allerdings werden nur Prozesse mit einem Kundennutzen als Geschäftsprozeß bezeichnet. Dies stellt eine für die Zwecke dieser Arbeit unnötige Einschränkung dar, denn die informationstechnische Unterstützung eines Prozesses ist unbeeinflußt davon, ob der Prozeß einen Kundennutzen hat oder nicht. Die Definition von [FeSi93] umfaßt auch Prozesse ohne Kundennutzen und ist daher besser für die Zwecke dieser Arbeit geeignet. In ihr wird der Geschäftsprozeß als Anordnung einer oder mehrerer Transaktionen zwischen betrieblichen Objekten gesehen. Im Rahmen dieser Transaktionen werden Leistungen und/oder Daten ausgetauscht. Diese Definition stimmt in hohem Maße mit der oben entwickelten Definition des Prozesses als strukturierter Interaktion überein. Über die vorangegangenen Definitionen des Begriffs Geschäftsprozeß hinaus werden auch interne Prozesse, die keinen direkten Kundennutzen bringen, erfaßt. Basierend auf der oben entwickelten Definition des Begriffs Prozeß kann eine eigene Definition entwickelt werden.

Definition 6 **Geschäftsprozeß**
Ein Geschäftsprozeß ist ein Prozeß in einem Wirtschaftsunternehmen oder einer Organisation.

Für die im Rahmen eines Geschäftsprozesses auszuführenden Schritte gilt, daß diese nicht auf eine betriebliche Funktionseinheit beschränkt sein müssen, sondern sich über funktionale, hierarchische und organisatorische Grenzen hinweg erstrecken können. Im letzteren Fall soll, wie in der Einleitung bereits definiert, von weitreichenden Prozessen die Rede sein.

Definition 7 **Weitreichender Geschäftsprozeß**
Ein weitreichender Geschäftsprozeß ist ein Geschäftsprozeß, dessen Prozeßschritte über mehrere funktionale und organisatorische Einheiten verteilt sind.

2.2.2 Geschäftsprozeßmodellierung

Vorrangiges Ziel der Geschäftsprozeßmodellierung [VoBe96] ist die Abbildung der betrieblichen Wirklichkeit und deren Optimierung, beispielsweise im Rahmen des Business Process Reengineering [HaCh93]. Die Workflow-Modellierung dagegen zielt auf die Optimierung der informationstechnischen Unterstützung von Prozessen durch Workflows. Allein um Zielkonflikte zu vermeiden, sind getrennte Modelle für Geschäftsprozesse und Workflows notwendig [Ambe97]. Gestützt wird dieses Vorgehen auch durch die Erfahrungen bei der Informationssystementwicklung in Betrieben. So wurde in der Vergangenheit das Ziel der Erfassung und Darstellung betrieblicher Prozesse mit dem der Informationssystementwicklung vermengt. Dies führte jedoch zu einer Reihe von Problemen, die in [Sinz98], [Sinz95a] dargestellt sind.

So hat die Herausbildung einer separaten fachlichen Modellierung der Prozesse eines Unternehmens nicht zuletzt ihre Ursache in den Erfahrungen mit den sogenannten Unternehmensdatenmodellen [Balz96]. Ziel von Unternehmensdatenmodellen ist die Beschreibung aller Informationen eines Unternehmens mit dem Ziel, eine einheitliche Sichtweise auf sie zu erreichen. Problematisch an ihnen ist jedoch die Konsistenzhaltung der auf verschiedenen Abstraktionsniveaus befindlichen Teilmodelle und der für Fachabteilungen mitunter schwierige Zugang zu den informationstechnisch ausgelegten Unternehmensdatenmodellen [Sinz95a].

Aus diesem Grund wurde in den letzten Jahren neben einer begrifflichen Unterscheidung von Geschäftsprozessen und Workflows auch eine gesonderte Modellierung beider Bereiche angestrebt. Einen ersten Schritt in diese Richtung macht die ARIS-Architektur [Sche94], die Geschäftsprozesse und –modelle in den Mittelpunkt stellt. Durch die Steuerungssicht werden die Daten-, Funktions- und Organisationssicht auf die Geschäftsprozesse unter einer fachlichen Perspektive zusammengeführt. Ereignisgesteuerte Prozeßketten bilden die Basis der Steuerungssicht. Sie enthalten Elemente der Funktionssicht und der Datensicht. Den Funktionen können wiederum Elemente der Organisationssicht zugeordnet sein. ARIS führt zwar eine Trennung von fachlicher und informationstechnischer Sicht durch, verzichtet jedoch auf unterschiedliche Abstraktionsebenen bei der Durchführung der fachlichen Modellierung. Hierdurch ist keine Komplexitätsreduktion auf fachlicher Seite möglich [Sinz95a], was bei umfangreichen Modellen leicht zur Unübersichtlichkeit führt.

2.2.2.1 Das Semantische Objektmodell (SOM)

Dieser Schwäche versucht das Semantische Objektmodell (SOM) [FeSi93], [FeSi94], [FeSi96b], [Sinz98] zu begegnen. Es soll hier vorgestellt werden, da es in späteren Kapiteln Verwendung finden wird. SOM besteht aus drei fachlichen Modellebenen: Unternehmensplan, Geschäftsprozeßmodellen und (betriebswirtschaftlichen) Ressourcen. Die Teilmodelle sind in Form von Metamodellen beschrieben. Durch ein Vorgehensmodell wird außerdem der Prozeß der Modellbildung geleitet. Es ist in Abbildung 6 dargestellt.

Unternehmensplan	Unternehmensziele- und strategien, Erfolgsfaktoren
Geschäftsprozesse	Lösungsverfahren für die Unternehmensziele
Ressourcen	Mittel für die Umsetzung der Lösungsverfahren

Abbildung 6: 3-Ebenen-Facharchitektur des SOM-Modells

Der Unternehmensplan entsteht als Ergebnis der betriebswirtschaftlichen Planung und legt grundsätzliche Unternehmensziele sowie die Strategien zu ihrer Erreichung fest. Er wird nur informell dargestellt. In ihm wird eine Abgrenzung zwischen dem Unternehmen und seiner Umwelt durchgeführt, und es werden die zwischen Beiden ausgetauschten Leistungen identifiziert. Dabei bezieht sich der Begriff Umwelt auf außerhalb des Unternehmens liegende Prozeßteilnehmer, nicht jedoch auf außerhalb der Prozeßsicht liegende Realitätsbereiche, wie beispielsweise juristische Abhängigkeiten zwischen Prozeßelementen. Eine Untersuchung von Realitätsbereichen außerhalb der Prozeßsicht wird in [Kuhn98] durchgeführt. Weitere Bestandteile sind Strategie und Erfolgsfaktoren.

Ein Ausschnitt aus einem hypothetischen Unternehmensplan ist in Abbildung 7 dargestellt. Ein Unternehmensziel ist die Gewinnmaximierung. Die Strategie zum Erreichen dieses Ziels sieht die Minimierung von Durchlaufzeiten sowie die Reduzierung von Lagerbeständen vor. Das zweite Unternehmensziel ist die Umsatzsteigerung. Dieses Ziel soll durch die Erhöhung der Kundenzufriedenheit erreicht werden.

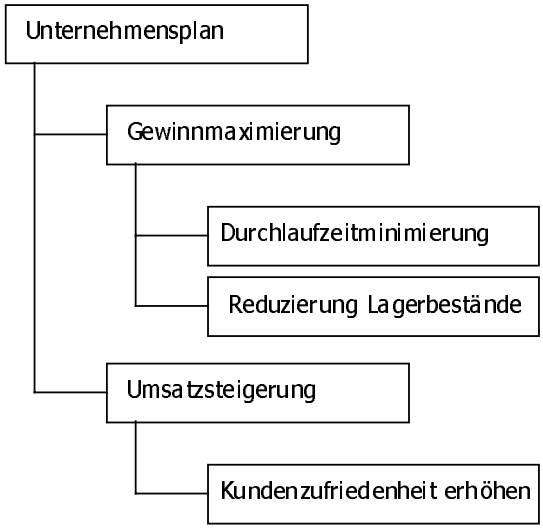


Abbildung 7: Unternehmensplan in SOM

Während also der Unternehmensplan die Außensicht der Prozesse wiedergibt, dient die Ebene der Geschäftsprozeßmodelle der Wiedergabe der Innensicht der Prozesse. Auf der Ebene der Geschäftsprozesse werden die Lösungsverfahren zum Erreichen der Ziele beschrieben, die auf der Unternehmensplanebene festgelegt worden sind. Dabei wird zwischen Haupt- und Serviceprozessen unterschieden. Hauptprozesse dienen direkt den auf der ersten Ebene definierten Zielen. Serviceprozesse erbringen für Haupt- oder Serviceprozesse Leistungen. Nicht bestimmt werden jedoch die dafür einzusetzenden Ressourcen, was auf der dritten Ebene geschieht. Zur

Darstellung der Geschäftsprozesse werden zwei Sichten verwendet, die struktur- und die verhaltensorientierte Sicht.

In der strukturorientierten Sicht, die in Abbildung 8 wiedergegeben ist, werden betriebliche Objekte und ihr Zusammenwirken über Transaktionen betrachtet. Ein Geschäftsprozeß besteht aus mindestens zwei betrieblichen Objekten und einer Transaktion. Betriebliche Objekte stellen Funktionseinheiten, Personen usw. dar. Sie können danach unterschieden werden, ob sie sich in der Diskurswelt oder deren Umwelt befinden. Unter Diskurswelt wird dabei der Teil der Realität verstanden, der im Rahmen der Modellierung erfaßt werden soll. Umweltobjekte dienen zur Darstellung von Objekten, die außerhalb dieser Diskurswelt liegen, aber zu denen dennoch Transaktionsbeziehungen bestehen. Transaktionen transportieren Leistungen oder Ereignisse. Der Transport von Ereignissen stellt die Verbindung zur verhaltensorientierten Sicht dar. Die Transaktionen in SOM unterstützen zwei Formen der Koordination: Die gleichberechtigte Koordination, die dem Verhandlungsprinzip folgt, und die hierarchische Koordination, wie sie als Folge beispielsweise von Weisungsbefugnissen entsteht und dem Regelungsprinzip gehorcht.

Bei der gleichberechtigten Koordination stehen zwei Objekte über drei Transaktionen in Verbindung. Durch die Anbahnungstransaktion T_a werden zunächst Informationen über die Vertragsbedingungen ausgetauscht. Falls diese Informationen bereits bekannt sind, kann die Anbahnungstransaktion auch entfallen. Ihr folgt die sogenannte Vereinbarungstransaktion T_v, an deren Ende die Festlegung der Leistungserbringung zwischen den Objekten steht. Auch die Vereinbarungstransaktion ist optional, d.h. sie kann entfallen, wenn die Vereinbarung bereits früher festgelegt wurde. Obligatorisch ist die Durchführungstransaktion T_d, die den eigentlichen Austausch der Leistung beschreibt.

Eine hierarchische Koordination kann allein aus einer Steuertransaktion T_s bestehen, oder es kann zusätzlich eine Kontrolltransaktion T_k vorhanden sein. Sowohl Steuer- und Kontroll-, als auch Anbahnungs-, Vereinbarungs- und Leistungstransaktionen können ihrerseits sowohl sequentiell als auch parallel weiter zerlegt werden. Hierdurch kann das Zusammenwirken der betrieblichen Objekte zunächst auf hohem abstraktem Niveau beschrieben und dann schrittweise verfeinert werden.

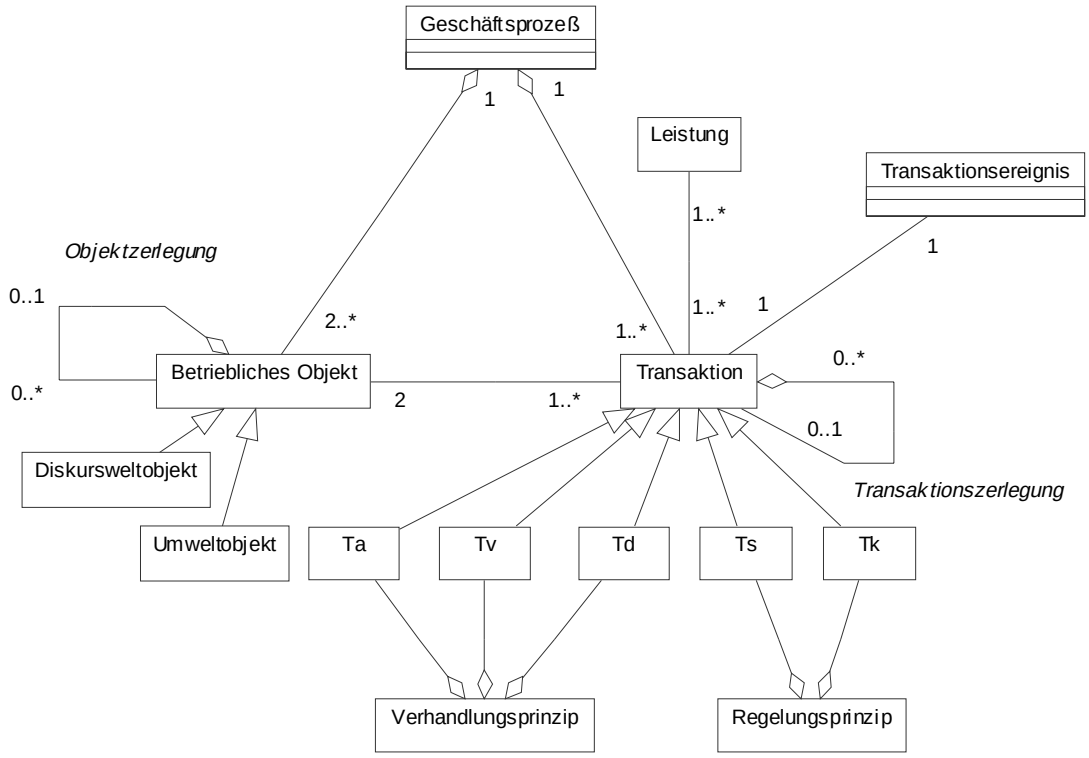


Abbildung 8: Strukturorientierte Sicht

Ergänzt wird die strukturorientierte Sicht durch sogenannte Interaktionsdiagramme. Ein Beispiel für ein Interaktionsdiagramm befindet sich in Abbildung 9. In ihm sind die drei betrieblichen Objekte Kunde, Vertrieb und Fertigung und die Transaktionen Auftrag und Fertigungsauftrag wiedergegeben. Da sich die weitere Darstellung auf den Vertrieb bezieht, ist der Vertrieb als Diskursweltobjekt in Form eines Rechteckes mit abgerundeten Ecken dargestellt. Kunde und Fertigung sind als Umweltobjekte dargestellt, erkennbar an den Ellipsen. Zwischen Kunde und Vertrieb besteht die Transaktion Auftrag, zwischen Vertrieb und Fertigung die Transaktion Fertigungsauftrag.

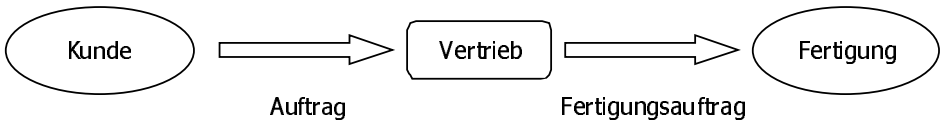


Abbildung 9: Interaktionsdiagramm

Die strukturorientierte Sicht ermöglicht durch die Anwendung der Objektzerlegung zunehmend feinere Darstellung der Geschäftsprozesse zu schaffen. So wird das Objekt Vertrieb in die Auftragserfassung und –prüfung zerlegt, wie in Abbildung 10 dargestellt. Zwischen den Objekten Auftragserfassung- und –prüfung befindet sich die Transaktion „erfaßter Auftrag“.

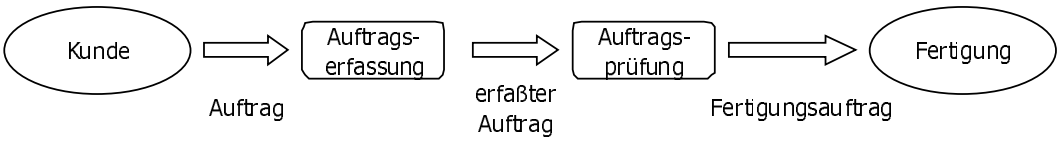


Abbildung 10: Verfeinertes Interaktionsdiagramm

Die zweite Sicht auf die Geschäftsprozesse eines Unternehmens im Rahmen des SOM-Modells ist die verhaltensorientierte Sicht, sie ist in Abbildung 11 dargestellt.

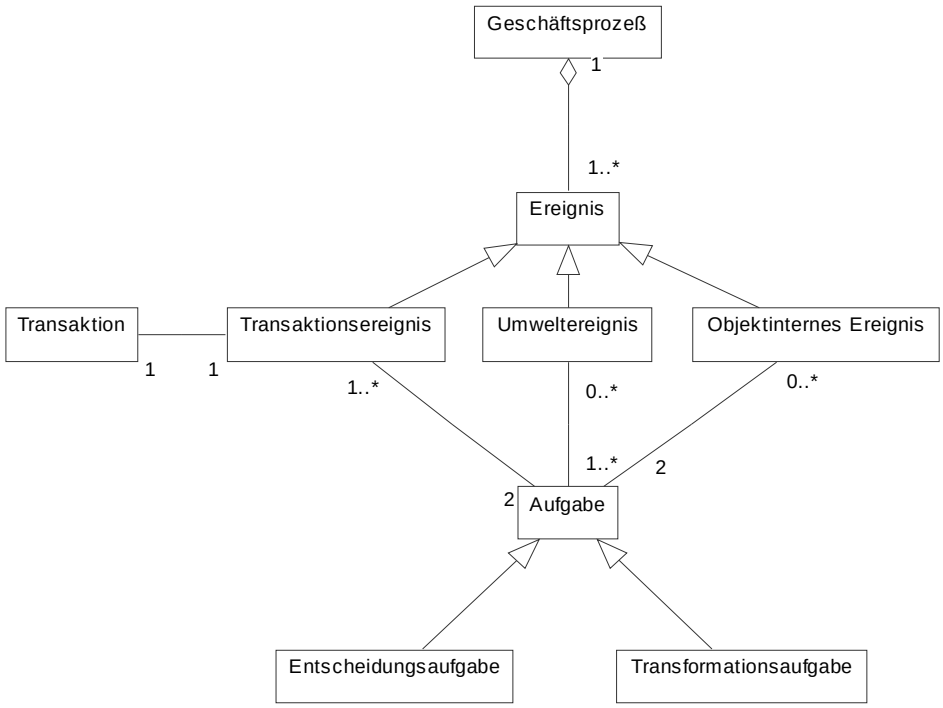


Abbildung 11: Verhaltensorientierte Sicht

In ihr wird der Ablauf der Geschäftsprozesse durch eine Abfolge von Aufgaben dargestellt, die über Ereignisse verknüpft sind. Bei den Aufgaben kann es sich um Entscheidungsaufgaben oder Transformationsaufgaben handeln. Entscheidungsaufgaben treffen auf Basis des bisherigen Verlaufs des Geschäftsprozesses Entscheidungen für dessen weiteren Verlauf. Transformationsaufgaben führen Operationen an den betrieblichen Objekten aus und können weiter zerlegbar sein.

In der verhaltensorientierten Sicht können drei Arten von Ereignissen auftauchen: Transaktionsereignisse, objektinterne Ereignisse und Umweltereignisse. Umwelt- und Transaktionsereignisse können die Rolle eines Vor- oder eines Nachereignisses einnehmen und so die Ausführung von Aufgaben steuern, was in Abbildung 12 veranschaulicht wird. Ein Vorereignis stößt eine Aufgabe an. Bei Abschluß der Aufgabe werden ein oder mehrere Nachereignisse erzeugt. Bei Transformationsaufgaben treten alle Nachereignisse ein, bei Entscheidungsaufgaben können keine, einige, oder alle Nachereignisse entstehen. Entscheidungsaufgaben können also durch die gezielte Erzeugung eines oder mehrerer Nachereignisse den Kontrollfluß in Geschäftsprozessen steuern, indem die auf sie folgende Aufgabe bestimmt wird. Startaufgaben, die am Anfang eines Geschäftsprozesses stehen, können durch Aufgaben ohne Vorereignisse dargestellt werden. Stopaufgaben am Ende eines Geschäftsprozesses werden durch Aufgaben ohne Nachereignisse wiedergegeben.

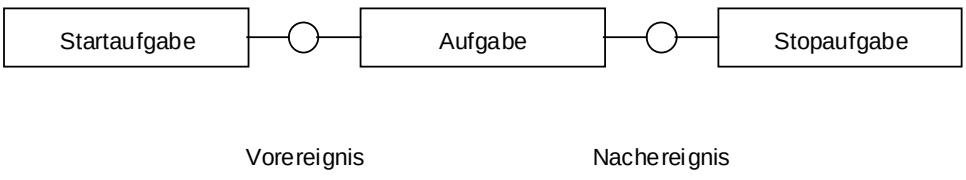


Abbildung 12: Ereignisse in SOM

Die verhaltensorientierte Sicht wird durch sogenannte Vorgangs-Ereignis-Schemata dargestellt. In Abbildung 13 ist ein derartiges Schema für das in Abbildung 10 durch Objektzerlegung erhaltene, verfeinerte Interaktionsdiagramm dargestellt. Die Objekte aus Abbildung 10 finden sich in Form von grau ausgefüllten Rechtecken wieder. Innerhalb der Objekte befinden sich Aufgaben, die durch ein Rechteck dargestellt werden und die über Transaktions- oder objektin-

terne Ereignisse verbunden werden. Ausgangspunkt ist das Objekt Kunde, dessen Aufgabe es ist, Aufträge zu erteilen, was sich in einem Transaktionsereignis „Auftrag erteilt“ niederschlägt. Das Objekt Auftragserfassung hat die Aufgabe „Auftrag erfassen“. In Abhängigkeit von der Höhe des erfaßten Auftrags tritt entweder das Ereignis „Auftragswert <= 5000“ oder das Ereignis „Auftragswert > 5000“ ein. Dementsprechend wird entweder eine Standard- oder eine Detailprüfung durchgeführt. Das Ergebnis ist entweder ein objektinternes Ereignis, das die Aufgabe Absage versenden anstößt, oder das Transaktionsereignis „Auftrag ok“, das die Weiterleitung des Auftrags an die Fertigung auslöst.

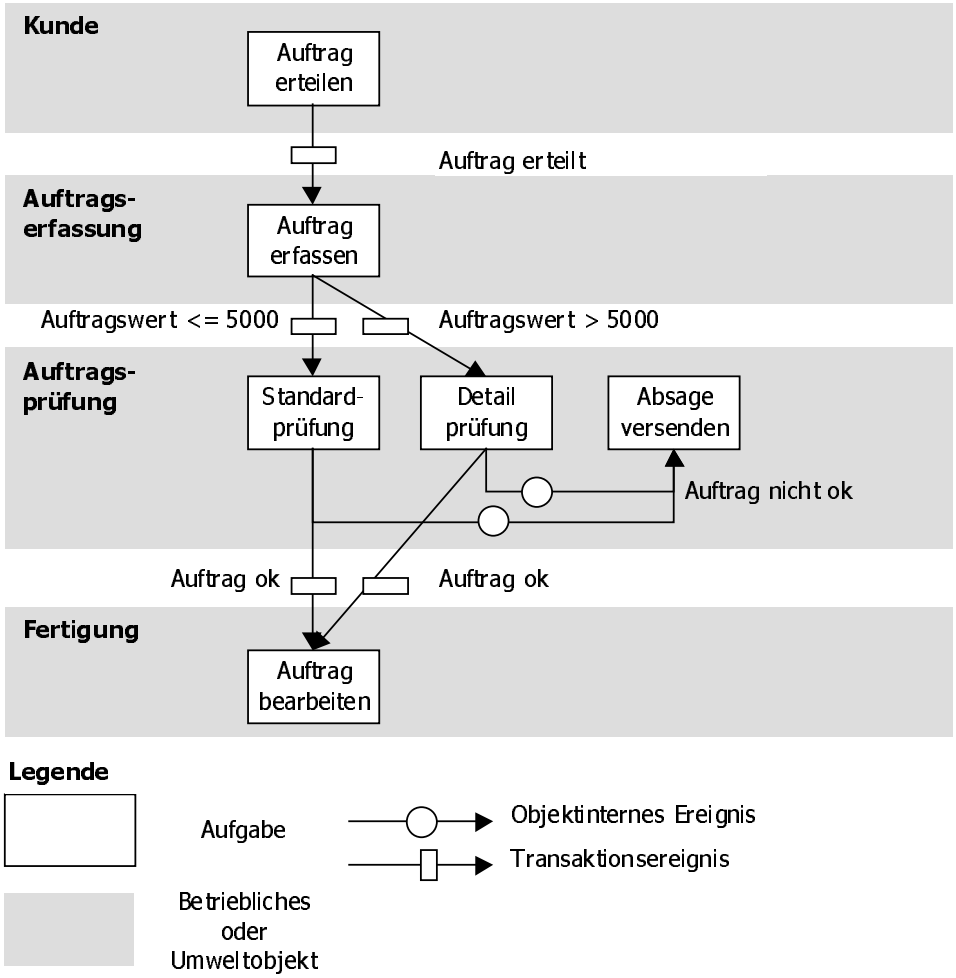


Abbildung 13: Vorgangs-Ereignis-Schema

Obwohl Geschäftsprozeßmodelle wie SOM einen deutlichen Fortschritt bei der Darstellung der betrieblichen Realität im Hinblick auf die Optimierung von Arbeitsabläufen darstellen, darf nicht vergessen werden, daß sie nur einen Ausschnitt aus der das betriebliche Geschehen beeinflussenden Realität darstellen. Auch hier wird wieder der Begriff des Modells als zielgerichtete Abbildung in Erinnerung gerufen werden. Bei Geschäftsprozeßmodellen ist das Modellziel die Erfassung, Darstellung und Optimierung prozeßartiger Abläufe im Unternehmen, keinesfalls aber die Gesamtdarstellung des Unternehmens. Daher existieren auch außerhalb der Geschäftsprozeßmodelle Realitätsbereiche, die Auswirkungen auf das Geschehen im Unternehmen und damit auch den Ablauf von Geschäftsprozessen haben können. Sie schlagen sich in Ereignissen nieder, die Abweichungen vom planmäßigen Ablauf eines Geschäftsprozesses ihre Ursache haben [Kuhn98]. Die Modellierung dieser Realitätsbereiche und deren informationstechnische Unterstützung stellen ein umfangreiches Gebiet dar, das den Umfang dieser Arbeit sprengen würde. Daher wird es hier nicht weitergeführt.

2.3 Workflows

Der Begriff Workflow wird häufig synonym zu dem des (Geschäfts-)Prozesses verwendet, wodurch eine Vermischung fachlicher und informationstechnischer Aspekte eintritt. Auch bei Trennung der Begriffe ist das Verhältnis zwischen (Geschäfts-)Prozeß und Workflow vielfach nicht klar definiert, wie das folgende Beispiel zeigt: So wird in [JaBS97] ein Workflow definiert als „.. eine, zum Teil automatisiert ablaufende, Gesamtheit von Aktivitäten, die sich auf Teile eines Geschäftsprozesses oder andere organisationelle Vorgänge beziehen“. Es wird offen gelassen, ob Workflows als informationstechnische Entsprechung eines Prozesses zu sehen sind oder nicht, und wie das fachliche Gegenstück zum Workflow beschaffen ist. Auf einer derart unscharfen Definition können natürlich nur schwer aussagekräftige Anforderungen an ein WfMS abgeleitet werden.

Eine Klärung des Verhältnisses von Workflows zu Geschäftsprozessen schafft die Definition von [GeHS95]: „We define a workflow as a collection of *tasks* organized to accomplish some business process (e.g., processing purchase orders over the phone, provisioning telephone service processing insurance claims)“. Konkreter noch ist die Definition der Workflow Management Coalition [WfMC]. Ein Workflow ist nach ihr „the automation of a business process, in whole or part, during which documents, information or tasks are passed from one participant to another for action, according to a set of procedural rules“. Ein Workflow wird hier eindeutig als informationstechnische Entsprechung eines Geschäftsprozesses gesehen. Das dieser Definition zugrundeliegende Verständnis eines Geschäftsprozesses deckt sich zudem gut mit der in dieser Arbeit gegebenen Definition eines Geschäftsprozesses. So finden sich sowohl der kooperative Charakter des Geschäftsprozesses durch die gemeinsame Bearbeitung von Daten, als auch der koordinierende Charakter durch die Angabe von Regeln für die Abfolge von Bearbeitungsschritten in der Definition des Workflow wieder. Basierend auf den obigen Definitionen sollen Workflows wie folgt definiert werden:

Definition 8 **Workflow**
Ein Workflow ist die informationstechnische Umsetzung eines Prozesses. Er beschreibt die funktionale Zerlegung der Aufgaben des Prozesses, die zu ihrer informationstechnischen Umsetzung verwendeten Ressourcen, sowie deren Zusammenspiel über Kontroll- und Datenflüsse.

Da sich Workflows als informationstechnisches Konzept explizit auf Prozesse als Urbild ihrer Abbildung beziehen, sind sie für eine ganzheitliche Prozeßunterstützung prädestiniert. Dabei ist jedoch ein grundsätzlich anderes Vorgehen als beispielsweise mit objektorientierten oder prozeduralen Ansätzen notwendig [JaBS97]. Auch gegenüber Datenbanken lassen sich Workflows abgrenzen. Während Datenbanken eine statische Sicht auf die Daten in Form von Datenmodellen haben, ist die Sicht eines Workflows dynamisch. D.h. in einem Workflow wird nicht der Zustand der Daten zu einem bestimmten Zeitpunkt, sondern die Veränderung der Daten im Lauf der Zeit durch die Operationen des Workflows betrachtet.

Die zur Unterstützung verschiedener Arten von Prozessen verwendeten Workflows lassen sich nach ihrem Gehalt an kooperativen und koordinativen Elementen unterscheiden [Moha96]. Workflows, die Entwicklungsprozesse umsetzen, sind meist stark kooperativ, die Bearbeitung der Entwurfsdaten steht im Vordergrund. Die Integration von kooperativen Workflows mit Gruppenarbeitsmechanismen wird in [ScLu95] behandelt. Dem gegenüber sind Verwaltungsprozesse meist sehr stark koordinativer Natur, die Einhaltung einer vorgegebenen Reihe von Arbeitsschritten steht im Vordergrund. Die kooperativ bearbeiteten Daten, also beispielsweise ein Antrag, sind bei weitem nicht so umfangreich wie bei Entwurfsvorgängen. Workflows, die der Unterstützung von Geschäftsprozessen dienen, stehen meist zwischen diesen beiden, indem

sie sowohl den Zugriff auf umfangreiche Datenbestände als auch umfangreiche Koordinationsmechanismen enthalten. Durch die Notwendigkeit, den Zugriff auf eine Vielzahl von Ressourcen in einer verteilten und heterogenen Umgebung zu gewährleisten, werden sie als eine der anspruchsvollsten Formen von Workflows angesehen [AAAM97]. Wissenschaftliche Workflows [WWVB96], [WVMe96], [SiMl96] ähneln in vielen Punkten Produktionsworkflows, was aber hier nicht weiter ausgeführt werden soll. Die Implementierung eines entsprechenden WfMS ist in [VWWi96], [VoWe98] beschrieben.

2.3.1 Workflow-Modelle und -Metamodelle

Die Trennung zwischen fachlicher Sicht im Rahmen von Geschäftsprozessen und informationstechnischer Sicht im Rahmen von Workflows wird auch bei der Modellierung von Geschäftsprozessen und Workflows durchgeführt. Vorrangiges Ziel der Geschäftsprozeßmodellierung ist die Abbildung der betrieblichen Wirklichkeit und deren Optimierung, beispielsweise im Rahmen des Business Process Reengineering [HaCh93]. Die Workflow-Modellierung dagegen zielt auf die Optimierung der informationstechnischen Unterstützung von Prozessen durch Workflows [Jabl95c], [ASWe96]. Allein um Zielkonflikte zu vermeiden, sind getrennte Modelle für Geschäftsprozesse und Workflows notwendig [Ambe97].

Wegen der weit verbreiteten homonymen Verwendung des Begriffs Modell sollen schrittweise die Begriffe der Workflow-Ausprägung, des Workflow-Schemas, -Modells und –Metamodells sowie des Aspekts entwickelt werden.

Ausgangspunkt ist die Beschreibung eines real ausgeführten Workflows, Workflow-Ausprägung genannt. Eine Workflow-Ausprägung umfaßt beispielsweise die Informationen bezüglich der Erfassung und Prüfung eines Auftrags Nr. 4711, mit einem Auftragswert von DM 400. Ebenso enthält sie Informationen über den konkreten Verlauf des Workflows, beispielsweise ob eine Standard- oder Detailprüfung durchgeführt wurde.

Definition 9 **Workflow-Ausprägung**
Eine Workflow-Ausprägung ist die Repräsentation der Ausführung eines Workflows.

Im Workflow-Schema wird von ausprägungsspezifischen Informationen abstrahiert, beispielsweise von der konkreten Nummer des Auftrags, seinem Wert und den dementsprechend durchgeführten Prüfungen. Statt dessen werden die prinzipiell möglichen Wege für die Bearbeitung des Auftrags festgelegt. Beispielsweise wird die Festlegung getroffen, daß ein Auftrag ab einem Auftragswert von DM 5000 einer Detailprüfung unterzogen werden muß, andernfalls einer Standardprüfung.

Definition 10 **Workflow-Schema**
Ein Workflow-Schema ist die abstrahierte Darstellung einer Menge von gleichartigen Workflow-Ausprägungen.

Als Darstellungsmittel für Workflow-Schemata werden wir in dieser Arbeit die graphische Sprache $EventFlow_L$ [ScSt98] verwenden. Ein auf erweiterten Petri-Netzen beruhender Ansatz ist in [Ober95] beschrieben. Die Elemente der $EventFlow_L$ Sprache sollen an Hand des Beispieles beschrieben werden. Es ist in Abbildung 14 dargestellt.

Jedes Workflow-Schema in $EventFlow_L$ beginnt mit einer grau gekennzeichneten Anfangsmarke und einer gleichfarbigen Endmarke. Die Arbeitsschritte sind als Rechtecke dargestellt. Das hier dargestellte Gabelungselement führt eine Alternativenauswahl durch, symbolisch dargestellt

durch das X für XOR im Gabelungselement. Durch die Angabe einer Bedingung, wie hier, daß der Auftragswert < 5000 sein muß, können Alternativen dargestellt werden. Ergänzt werden die Gabelungselemente durch die Verbindungselemente, die die gleiche Notation besitzen. Auch sie können durch Bedingungen ergänzt werden.

Ausgetauschte Informationen werden durch Dokumente repräsentiert, die mit den Arbeitsschritten verbunden sind. Die Richtung des Informationsflusses wird durch Pfeile angegeben. Das Rechteck an der Verbindungslinie zwischen Auftrag und Auftrags-erfassung zeigt an, daß die Information in diesem Arbeitsschritt neu geschaffen wurde. Der Status der Dokumente wird in Klammern unterhalb des Dokumenttyps angegeben. Durch ihn ist es möglich einen Datenfluß anzugeben, der sich vom Kontrollfluß unterscheidet.

Unterhalb der Arbeitsschritte können Programme und Rollen angegeben werden, die die Ausführung der einzelnen Arbeitsschritte übernehmen sollen. Durch die Verwendung von Rollen wird also nicht direkt ein Mitarbeiter zugewiesen, sondern es ist möglich, zur Laufzeit des Workflows einen Mitarbeiter zuzuweisen. Im Beispiel sind dem Arbeitsschritt Auftragserfassung das Programm „Auftragserfassung“ und ein Sachbearbeiter, dem Arbeitsschritt detaillierte Eingangsprüfung das Programm „Detaillierte Prüfung“ und ein Abteilungsleiter, sowie dem Arbeitsschritt Standardprüfung das Programm „Standardprüfung“ und ein Sachbearbeiter zugeordnet.

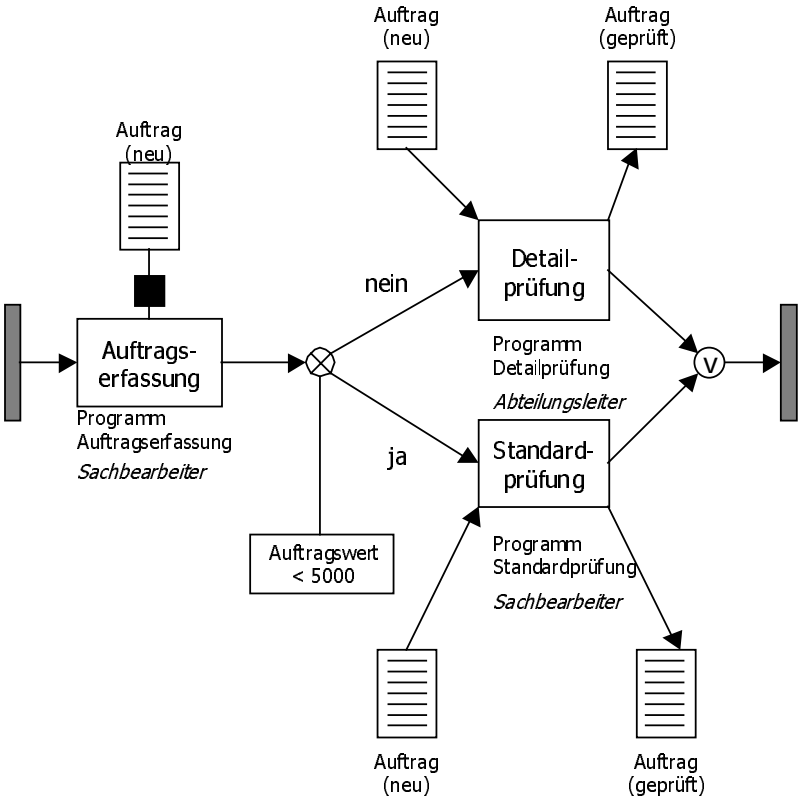


Abbildung 14: Darstellung des Beispielworkflows in $EventFlow_L$

Genauso wie das Workflow-Schema von einer Menge von Entitäten abstrahiert hat, abstrahiert das Workflow-Modell von einer Menge von Workflow-Schemata. In einem anderen Schema könnte der Parameter des XOR-Gabel-Elements Angebotswert und der Entscheidungswert DM 10000 betragen. Das Workflow-Modell abstrahiert also von der spezifischen Verwendung eines Schemaelementes und beschreibt die Menge der zur Bildung von Schemata zur Verfügung stehenden Elemente. Diese könnte außer dem XOR-Gabel-Element auch ein UND-Gabel-Element enthalten, das das parallele Durchlaufen zweier Bearbeitungswege fordert.

Definition 11 **Workflow-Modell**
Ein Workflow-Modell beschreibt die Menge der Elemente, die zur Bildung von Workflow-Schemata zur Verfügung stehen.

Die in einem Workflow-Modell beschriebene Menge umfaßt Elemente für die Darstellung der funktionalen Zergliederung von Workflows, der zur Umsetzung verwendeten informationstechnischen Ressourcen sowie des Zusammenspiels über Kontroll- und Datenflüsse. Darüber hinaus kann auch die Notwendigkeit nach weiteren Darstellungsmitteln entstehen. Die von einem Workflow-Modell bereitzustellenden Darstellungsmittel sind also deutlich umfangreicher als die eines objektorientierten oder –relationalen Modells. Daher empfiehlt es sich, die Gestaltung des Workflow-Modells einer Systematik zu unterwerfen. Wie in Abschnitt 2.1.1 dargelegt, geschieht dies in Form eines Workflow-Metamodells.

Definition 12 **Workflow-Metamodell**
Ein Workflow-Metamodell umfaßt die Menge der Elemente, die zur Darstellung von Workflow-Modellen zur Verfügung stehen.

Workflow-Metamodelle, wie beispielsweise in [RoMü96], [DeVÖ96], beschreiben die Elemente von Workflow-Modellen und ihre Beziehungen. Ein systematischer Ansatz wird in [CuKO92], [Jabl95a] verfolgt. Ausgangspunkt ist die Identifikation von Aspekten. Mit ihrer Hilfe kann ein Workflow-Modell in disjunkte Mengen von Modellelementen unterteilt werden, die zur Darstellung sich unabhängig fortentwickelnder Bereiche der Realwelt verwendet werden. Beispielsweise entwickeln sich die koordinativen Elemente eines Prozesses unabhängig von den einzelnen Operationen. Sie werden daher in einem Workflow-Metamodell als zwei getrennte Aspekte wiedergegeben, nämlich als Verhaltens- und als Operationsaspekt.

Definition 13 **Aspekt**
Aspekte beschreiben Mengen von Elementen eines Workflow-Modells, mit denen sich orthogonale Gesichtspunkte der IT-Unterstützung von Prozessen darstellen lassen.

Auf der Basis von Aspekten ist das in [Jabl95a] und [JaBS97] beschriebene und in Abbildung 15 dargestellte Workflow-Metamodell entwickelt worden. Es ist in der UML-Notation [UML] dargestellt, wie auch die übrigen Klassendarstellungen dieser Arbeit. Das Workflow-Metamodell unterscheidet im wesentlichen Funktions-, Verhaltens-, Informations-, Organisations- und Operationsaspekt. Der Funktionsaspekt untersucht die Frage, was in einem Workflow ausgeführt werden soll und wie sich der gesamte Workflow aus Teil-Workflows und elementaren Workflows zusammensetzt. Ein Workflow, der aus untergeordneten Workflows besteht, wird als *kompositer* Workflow bezeichnet. Untergeordnete Workflows können selbst wieder komposit sein. *Elementare* Workflows sind Workflows, die keine untergeordneten Workflows enthalten. Der Verhaltensaspektbetrachtet die Steuerung der Abfolge der (Teil-)Workflows durch Anwendung von Kontrollstrukturen. Durch den Organisationsaspekt wird eine Verbindung zwischen dem Workflow und der Organisationsstruktur des Unternehmens hergestellt. Die Umsetzung von elementaren Workflows durch Workflow-Applikationen wird im Operationsaspekt untersucht. Der Informationsaspekt gibt den Datenaustausch zwischen den Teil-Workflows wieder. Über diese Aspekte hinaus werden zusätzliche Aspekte identifiziert, die zwar die Qualität eines Workflows beeinflussen, jedoch keine zwingende Voraussetzung für dessen Ausführbarkeit sind. Hierzu gehört beispielsweise der historische Aspekt, der die Ausführungshistorie von Workflows betrachtet und Grundlage des Workflow-Monitoring ist.

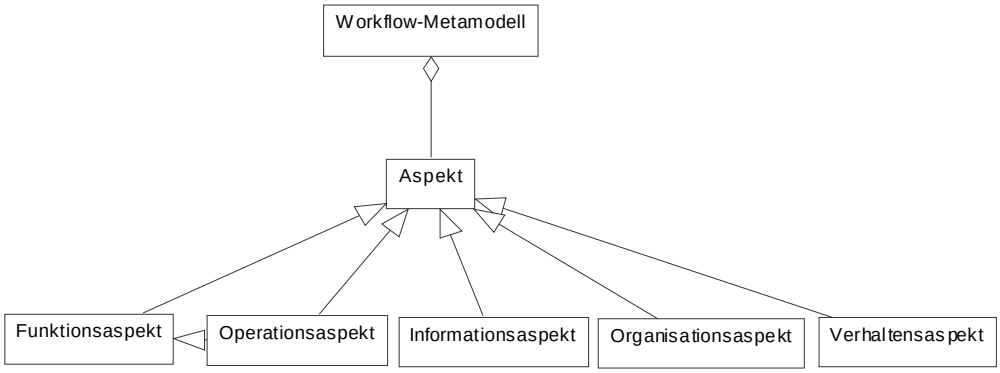


Abbildung 15: Workflow-Metamodell nach [Jabl95a]

Zusammenfassend bestehen zwischen Workflow-Metamodell, -Modell, -Schema und –Ausprägung folgende Zusammenhänge. Sie sind in Abbildung 16 dargestellt

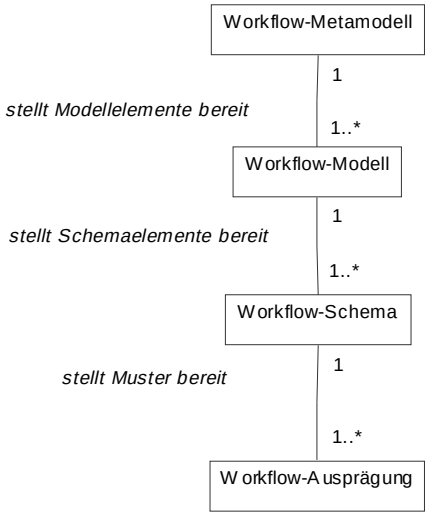


Abbildung 16: Workflow-(Meta)-Modell, -Schema- und Ausprägung

Diese Zusammenhänge werden an Hand eines Beispieles verdeutlicht. Dazu sollen Schritt für Schritt die Schemaelemente des in Abbildung 14 dargestellten Workflow-Schemas in ein Workflow-Modell eingeordnet werden. Das Workflow-Modell des Beispiels ist in Abbildung 17 dargestellt. Dabei werden nur die für die Darstellung des Workflow-Schemas benötigten Modellelemente dargestellt, obwohl das vollständige Modell umfangreicher ist. Wichtig ist also zu beachten, daß es sich dabei nicht um die Darstellung des Workflow-Schemas handelt, sondern nur der zu seiner Darstellung verwendeten Elemente des Workflow-Modells.

Die Darstellung der Aspektzugehörigkeit von Modellelementen ist mit den Mitteln der UML nur umständlich möglich. Ein Modellelement ist eine Ausprägung eines Aspektelementes, das wiederum im Rahmen einer Aggregationsbeziehung einem Aspekt zugehörig ist. Die Darstellung dieses Zusammenhangs führt sehr schnell zu unübersichtlichen Diagrammen. Daher wird in dieser Arbeit die Aspektzugehörigkeit der Modellelemente als Aggregationsbeziehung zum Aspekt wiedergegeben.

Ausgangspunkt ist der Arbeitsschritt Auftragserfassung, in dem das Programm Auftragserfassung durch einen Sachbearbeiter bedient wird und ein neues Auftragsdokument geschaffen wird. Die Spezifikation des Arbeitsschrittes ist ein Element des Funktionsaspekts, während das Programm zu seiner Umsetzung dem Operationsaspekt zugehörig ist. Das erzeugte Auftragsdokument gehört zum Informationsaspekt, die Rollenspezifikation "Sachbearbeiter" ist Bestandteil des Organisationsaspekts. Die nach dem Arbeitsschritt Auftragserfassung zugehöri-

ge Entscheidung, ob der Auftrag einer Detail- oder nur einer Standardprüfung unterzogen wird, stellt ein Element des Verhaltensaspekts dar. Es handelt sich um ein Gabelungselement, das entscheidet, ob eine Detailprüfung oder eine Standardprüfung des Auftrags durchgeführt wird. Kriterium für die Entscheidung ist, ob der Auftragswert < 5000 DM beträgt oder nicht. Zur einfacheren Bezeichnung sollen derartige Elemente als XOR-Gabel bezeichnet werden. Den Elementen des Funktionsaspekts Detailprüfung und Standardprüfung sind wiederum Elemente des Operationsaspekts in Form der Programme "Detailprüfung" und "Standardprüfung" zugewiesen. Die Rollenzuweisungen Abteilungsleiter und Sachbearbeiter sind Elemente des Organisationsaspekts. Den Abschluß bildet wiederum ein Element des Verhaltensaspekts. Es handelt sich um eine ODER-Verbindung.

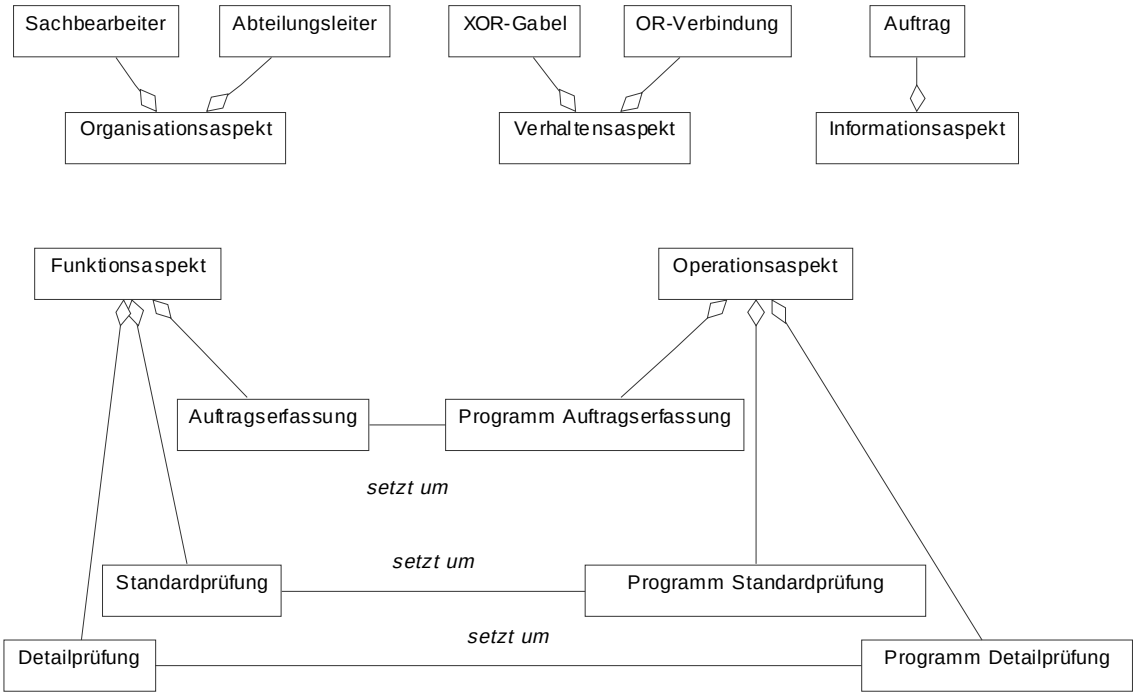


Abbildung 17: Workflow-Modell des Beispiels

2.3.2 Workflow-Management-Systeme

In [AAAM97], [Geor94] werden Workflow-Management-Systeme (WfMS) als eine Sammlung von Werkzeugen zur Erstellung und Definition, zur Festlegung der Ablaufumgebung und zur Beschreibung von Benutzerschnittstellen und verwendeten externen Anwendungen verstanden. Die Definitionen in [Rein93], [Jabl95a], [Jabl95b], [JaBu96] sehen in WfMS ein (re-)aktives Basissoftwaresystem zur Steuerung des Arbeitsflusses nach der Vorgabe einer Ablaufspezifikation, dem Workflow-Schema. Das WfMS dient als Grundlage für die Entwicklung von sogenannten Workflow-Management-Anwendungen (WfMA) und steuert und überwacht deren Ausführung. Eine Klassifikation verschiedener Formen von WfMS wird in [ScBö94] vorgenommen.

Definition 14 **Workflow-Management-Systeme**
Workflow-Management-Systeme sind Software-Systeme, die der Definition, Verwaltung, Ausführung, Überwachung und Protokollierung von Workflows dienen.

2.3.2.1 Phasen eines WfMS

Das Zusammenwirken der bisher vorgestellten Konzepte von der Festlegung eines Workflow-Modells bis zur Ausführung der Workflow-Ausprägung in existierenden WfMS wird an Hand von Abbildung 18 veranschaulicht.

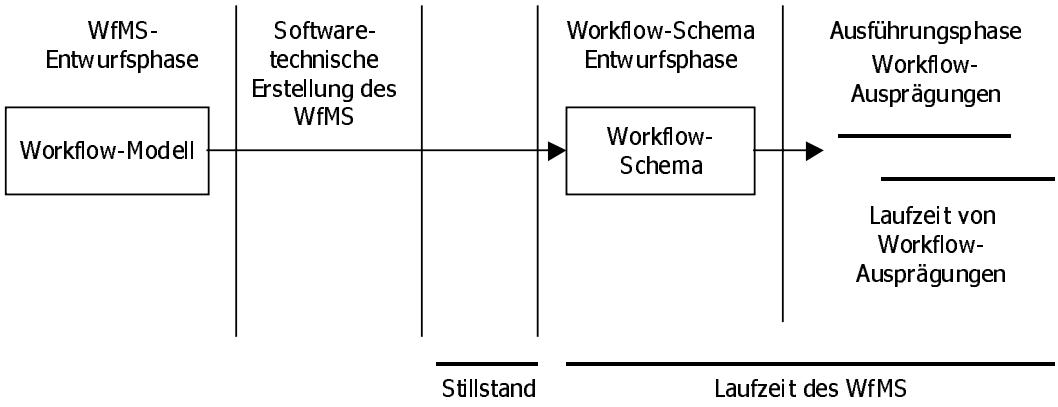


Abbildung 18: Phasen im WfMS

Ausgangspunkt ist die Festlegung des Workflow-Modells in der Entwurfsphase des WfMS. Mit ihm wird die Menge der Schemaelemente festgelegt, die vom WfMS verarbeitet werden können. Dieses Workflow-Modell wird als das Workflow-Modell des WfMS bezeichnet. In der Phase der Software-technischen Umsetzung wird das WfMS implementiert. Nach Abschluß dieser Phase kann es zwischen den Zuständen „im Stillstand“ oder „zur Laufzeit“ wechseln. Ist das WfMS im laufenden Zustand, kann es neue Workflow-Schemata aufnehmen. Diese Phase wird als Workflow-Schema-Entwurfsphase bezeichnet. In ihr werden auf der Basis des Workflow-Modells Workflow-Schemata gebildet, die reale Workflows unterstützen sollen. Diese Workflow-Schemata stehen am Ende der Entwurfsphase für die Schaffung von Workflow-Ausprägungen bereit. In der Ausführungsphase werden Workflow-Ausprägungen erzeugt und proaktiv ausgeführt.

2.3.2.2 Verteilung von WfMS

Ein wichtiges Unterscheidungsmerkmal für Workflow-Management-Systeme ist, ob und wie sie gegebenenfalls eine Verteilung des Workflows unterstützen. Zunächst wird zwischen der Verteilung und der Partitionierung unterschieden. Die Verteilung ist eine Eigenschaft des WfMS, die Partitionierung ist hingegen eine Eigenschaft von Workflow-Schemata. Letztere definiert sich als die Aufteilung eines Workflow-Schemas auf mehrere Partitionen durch den Modellierer. Diese Partitionen können dann auf verschiedenen WfMS ausgeführt werden, die für sich jedoch voll zentralisiert arbeiten können. Dargestellt ist dies in Abbildung 19. Im Gegensatz hierzu bezeichnet die Verteilung, wie sie in dieser Arbeit verstanden wird, eine Eigenschaft der Architektur des WfMS. Sie bezeichnet die Fähigkeit eines WfMS, die von ihm unterstützten Workflows an mindestens zwei Orten auszuführen.

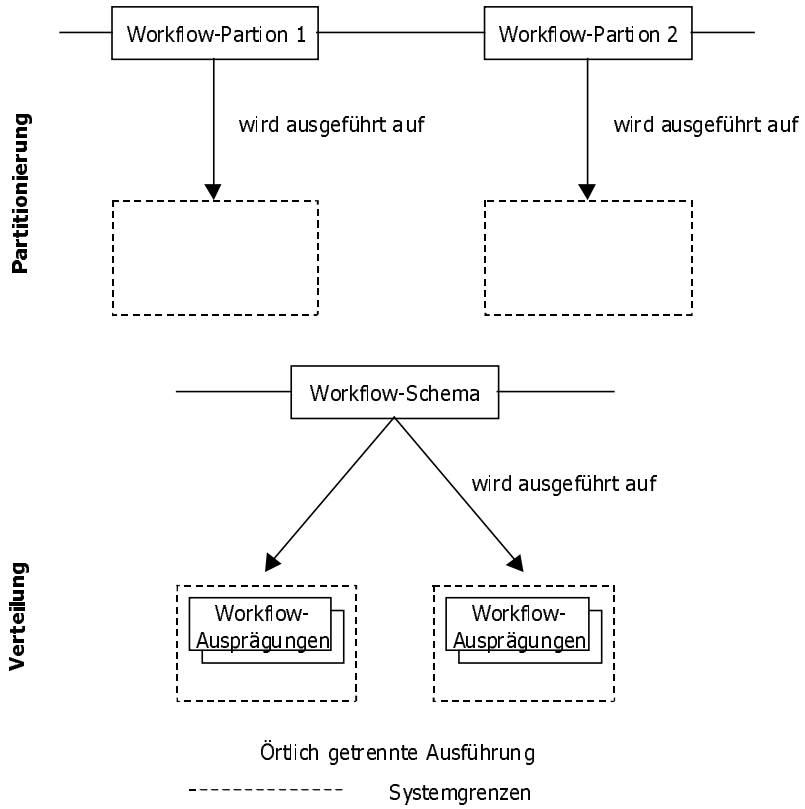


Abbildung 19: Partitionierung vs. Verteilung

Bisherige Untersuchungen zum Grad der Verteilung in WfMS [Schu97b], [BaDa98], [Baue98], [Schu98], [Baue99] erbringen keine für die Zwecke dieser Arbeit taugliche Kategorisierung. Daher soll hier das sogenannte *Ausführungsmodell* eingeführt werden. Es beschreibt, ob die in einem WfMS gebildeten Workflow-Ausprägungen an einem oder an mehreren Orten ausgeführt werden.

Es lassen sich drei Arten von Ausführungsmodellen unterscheiden, wie an Hand von Abbildung 20 verdeutlicht wird. Betrachtet werden zwei Workflow Schemata, A und B. Bei der zentralisierten Ausführung (a) werden alle Ausprägungen aller Schemata auf einem WfMS ausgeführt. Bei der schemaverteilten Ausführung werden die Ausprägungen verschiedener Schemata verteilt, während die Ausprägungen eines Schemas zentralisiert ausgeführt werden (b). Man spricht in diesem Zusammenhang auch von *Schemaverteilung*.

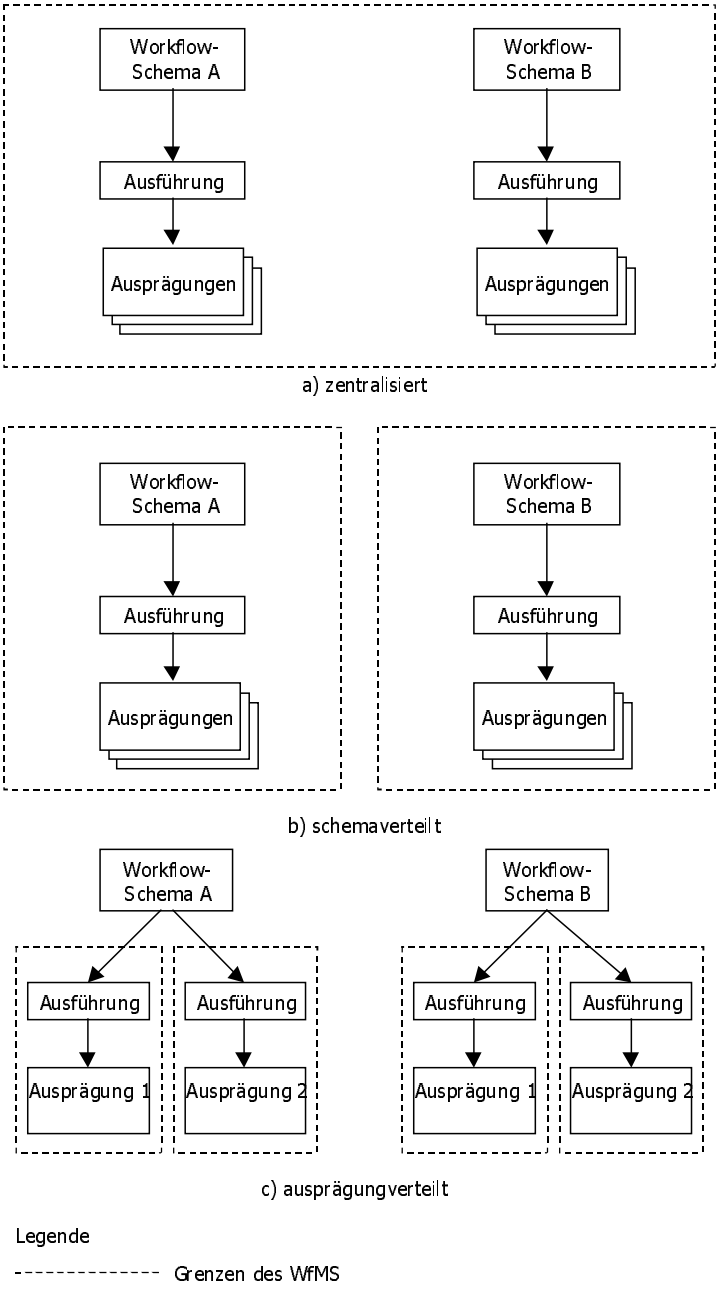


Abbildung 20: Ausführungsmodelle

Definition 15 **Schemaverteilung**
Fähigkeit eines WfMS, unterschiedliche Schemata an verschiedenen Orten auszuführen.

Ein noch höherer Grad an Verteilung ist erreichbar, wenn man nicht mehr nur verschiedene Schemata, sondern auch die einzelnen Ausprägungen der Schemata verteilt: Die *Ausprägungsverteilung* führt die Ausprägungen eines Workflow-Schemas verteilt aus (c).

Definition 16 **Ausprägungsverteilung**
Fähigkeit eines WfMS, Ausprägungen eines Schemas an verschiedenen Orten auszuführen.

2.4 Software-Systeme

In diesem Abschnitt sollen die Begriffe Software-Architektur, Software-System, Dienst, software-technische Ressource und Anwendung definiert werden. Dies geschieht, um eine terminologische Grundlage für die Beschreibung der in späteren Kapiteln identifizierten nichtfunktionalen Anforderungen aus der Unterstützung weitreichender Prozesse zu schaffen.

Ein Software-System besteht aus einer Menge von Ressourcen, die Dienste bereitstellen. Diese Dienste werden in Anwendungen miteinander verknüpft um einen bestimmten Zweck zu erfüllen. Die Regeln für die Verknüpfungen zwischen den Bestandteilen eines Software-Systems werden im Rahmen einer Software-Architektur festgelegt. Dieser Zusammenhang ist in Abbildung 21 visualisiert.

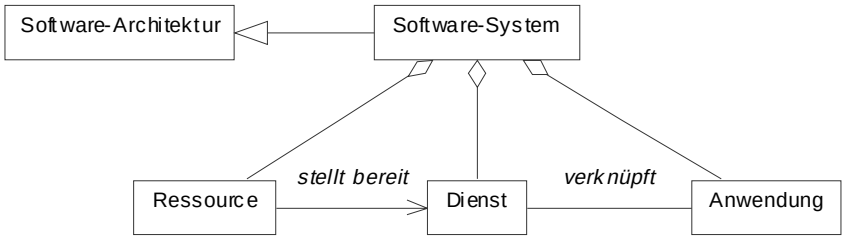


Abbildung 21: Software-System

Nun sollen die Bestandteile eines Software-Systems, also Ressourcen, Dienste und Anwendungen, näher betrachtet werden.

Definition 17 **Ressource**
Eine software-technische Ressource stellt einen oder mehrere Dienste bereit.

Ressourcen können je nach Software-Architektur in unterschiedlichen Formen auftreten. Beispiele sind Module, Objekte oder Komponenten. Die Gestaltung der Ressourcen beeinflußt stark die Eigenschaften des Software-Systems als Ganzem, beispielsweise die Fähigkeit, Ressourcen hinzuzufügen oder zu konfigurieren. Minimalanforderung zur Darstellung ist die Angabe einer Schnittstelle. Dienste können in ihren Eigenschaften angepaßt werden.

Definition 18 **Dienst**
Ein Dienst ist die Spezifikation einer Leistung.

Eine Anwendung „wendet Dienste an“ um einen Zweck zu erfüllen. Dazu verknüpft sie die Dienste und paßt sie für den zu erfüllenden Zweck an. In der Praxis fällt der Begriff der Anwendung häufig mit dem des Programms zusammen. Unter einem Programm versteht man eine software-technische Ressource, die eine Anwendung unterstützt. Sie enthält dazu Dienste in verknüpfter Form, die nicht durch andere Anwendungen genützt werden können.

Definition 19 **Anwendung**
Eine Anwendung ist eine zweckgebundene Verknüpfung und Anpassung von Diensten.

Für die Definition des Begriffs Rahmenwerk wird die Begriffsbildung von [Zimm97], [Rüpi97], übernommen, die für Rahmenwerk noch den Begriff „Framework“ verwendeten.

Definition 20 **Rahmenwerk**
Ein Rahmenwerk ist ein abstrakter Bauplan zur Konstruktion von Anwendungssystemen.

Ein Rahmenwerk besteht aus abstrakten und konkreten Bestandteilen[3], wie Abbildung 22 zeigt. Dabei handelt es sich typischerweise um Klassen, es sind aber auch andere Bestandteile möglich, wie beispielsweise Komponenten. Die Bestandteile eines Rahmenwerks können konkret oder auch abstrakt sein, wobei die konkreten Bestandteile Konkretisierungen der abstrakten Bestandteile sind. Rahmenwerke können danach unterschieden werden, in wie weit sie konkrete Bestandteile enthalten. Abstrakte Rahmenwerke enthalten nur abstrakte Bestandteile, während konkrete Rahmenwerke auch konkrete Bestandteile enthalten.

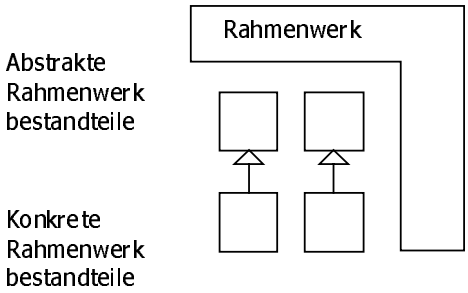


Abbildung 22: Symbolische Darstellung von Rahmenwerken

Das Vorgehen zur Herleitung eines Anwendungssystems heißt *Ausprägungsbildung* des Rahmenwerks. *Komposition* bedeutet, diejenigen abstrakten Bestandteile des Rahmenwerks auszuwählen, die für das Anwendungssystem benötigt werden. Einige Bestandteile werden von jeder Anwendung benötigt, andere Bestandteile sind nur in einigen Anwendungsvarianten enthalten. Die *Konkretisierung* besteht darin, Implementierungen zu allen abstrakten Bestandteilen zu finden. Bei einem objektorientierten Rahmenwerk gehört hierzu die Auswahl konkreter Klassen, die Entwicklung von konkreten Unterklassen sowie die Auswahl generischer Klassen und die Konkretisierung ihrer generischen Typparameter.

2.5 Zusammenfassung

In diesem Kapitel wurden die begrifflichen Grundlagen für die weitere Arbeit gelegt. Zunächst wurden die Begriffe Modell, Metamodell, Schema und Ausprägung definiert. Ein Modell stellt nach der hier entwickelten Definition eine zielgerichtete Abstraktion dar. Die Zielorientierung führt zu einer Perspektive auf das abzubildende System, die sich in entsprechend gebildeten Abstraktionen niederschlägt. Ein Schema stellt einen Spezialfall eines Modells dar, bei dem das abzubildende System reale Entitäten sind.

Auf diesen Grundlagen wurden dann Definitionen der Begriffe Prozeß, Geschäftsprozeß und Workflow, sowie mit ihnen in Verbindung stehender Begriffe geschaffen. Es wurde eine klare Trennung zwischen dem Prozeß als fachlicher Perspektive und dem Workflow als seiner informationstechnischen Entsprechung durchgeführt. Prozesse wurden von anderen Formen der Interaktion abgegrenzt und Workflows von anderen informationstechnischen Konzepten unterschieden.

[3] In [Rüpi97], [Zimm97] ist von nicht näher definierten Komponenten die Rede. Um Verwechslungen mit der später ausführlich dargestellten Komponente zu vermeiden, wird hier das Wort „Bestandteile“ verwendet.

Den Abschluß bildete die Definition des Begriffes Software-System sowie die Darstellung der Bestandteile von Software-Systemen. Ebenfalls wurde für den Begriff des Rahmenwerks eine Definition gegeben.

3 Szenario

In diesem Kapitel werden zunächst die in der Einleitung identifizierten Rahmenbedingungen für die umfassende Unterstützung weitreichender Prozesse an Hand eines Szenarios praxisnah dargestellt. Dies wird mit Hilfe eines Szenarios, das aus dem Projekt VORREITER (*V*irtuelle *OR*ganisation – optimaler Prozeßablauf durch *E*ntwicklung innovativer *I*nformations*T*echnologien, *E*rprobung und *R*e-engineering) stammt, geschehen. Es wurde gewählt, da virtuelle Unternehmen besonders deutlich die in der Einleitung identifizierten aufgestellten Rahmenbedingungen aufweisen. Die praktische Relevanz kann daher besonders gut exemplarisch belegt werden. Danach werden aus den Rahmenbedingungen für die Unterstützung weitreichender Prozesse informationstechnisch orientierte Anforderungen abgeleitet.

3.1 Beschreibung

Ziel des Projekts VORREITER ist es, die Zusammenarbeit mehrerer klein- und mittelständischer Unternehmen (KMUs) im Rahmen von virtuellen Unternehmen durch eine geeignete informationstechnische Unterstützung zu ermöglichen. Die am VORREITER Projekt beteiligten Unternehmen stellen Hard- und Software für Maschinensteuerungen her. Jedes Unternehmen für sich kann nicht alle für die Herstellung einer Maschinensteuerung notwendigen Schritte durchführen. Durch die Unterstützung eines virtuellen Unternehmens sollen die an VORREITER beteiligten Firmen in die Lage versetzt werden, auch als Anbieter für komplette Maschinensteuerungen aufzutreten und Aufträge annehmen zu können. Dies geschieht durch die Bündelung ihrer Kompetenzen, die in der Durchführung von Einzelschritten beim Bau einer Maschinensteuerung liegen.

Zu den am VORREITER Projekt gehörigen Unternehmen zählen: Ein Händler für den Vertrieb und Logistik von elektronischen Bauelementen, ein Modulhersteller, ein Platinenproduzent, ein Softwarehersteller, sowie ein Systemintegrator, der für den Einbau der kompletten Steuerungen in die Maschinen zuständig ist. Ihre Beiträge zum virtuellen Unternehmen sind in Abbildung 23 wiedergegeben.

Ausgangspunkt des Herstellungsprozesses der Maschinensteuerungen ist die Fertigung von sogenannten Multi-Chip-Modules (MCM) durch den Modulhersteller. MCMs bestehen aus ca. 60 Einzelbauteilen, die vom Bauelementehändler bezogen werden. Die nächste Stufe des Herstellungsprozesses führt ein Platinenhersteller durch, der die MCMs auf von ihm gefertigten Platinen zusammenführt, um eine komplette Maschinensteuerung zu bauen. Es fehlt allerdings noch die Steuerungssoftware. Sie wird von einem weiteren Unternehmen entwickelt und in den auf den Platinen befindlichen Speichern abgelegt wird. D.h. es werden EEPROMs, oder ähnliches, programmiert und auf den Platinen plaziert. Mit der Steuerungssoftware ausgestattete Platinen werden dann an den letzten Teilnehmer des virtuellen Unternehmens weitergegeben, einen Systemintegrator, der den Einbau der Platinen in die Maschine übernimmt.

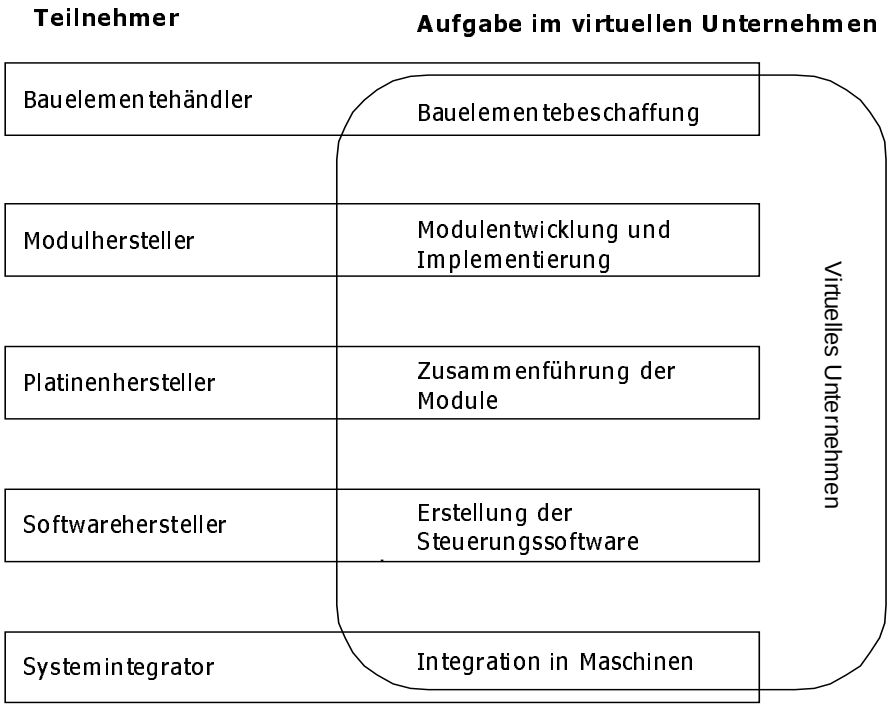


Abbildung 23: Virtuelles Unternehmen für die Herstellung von Maschinensteuerungen

Zur Darstellung eines virtuellen Unternehmens eignen sich prozeßorientierte Konzepte viel besser als eine statische, isolierte Sichtweise, um die an dem virtuellen Unternehmen teilnehmenden Personen und Ressourcen zu identifizieren und darzustellen.

Von den Geschäftsprozessen des virtuellen Unternehmens wird die Auftragsbearbeitung herausgegriffen, die in Abbildung 24 in der SOM-Notation dargestellt ist. Dies zeigt auch den Einsatz unterschiedlicher Modelle mit unterschiedlichen Perspektiven. Da momentan eine betriebswirtschaftliche Perspektive verfolgt wird, findet dementsprechend auch ein betriebswirtschaftlich orientiertes Modell wie SOM Verwendung. Wenn später eine stärker informationstechnisch orientierte Perspektive verfolgt wird, kommt die EventFlow-Notation zum Einsatz.

Gegenstand der Auftragsbearbeitung ist die Entgegennahme und Prüfung von Aufträgen an das virtuelle Unternehmen. Ausgangspunkt ist die Auftragserfassung. Dabei wird der in Papierform eingegangene Auftrag elektronisch erfaßt. Zunächst wird er einer Eingangsprüfung unterzogen, bei der die grundsätzliche Korrektheit des Auftrages geprüft wird. Dies wird stellvertretend für die übrigen Teilnehmer durch den Systemintegrator durchgeführt. Er ist in Abbildung 24 als betriebliches Objekt wiedergegeben. Die Eingangsprüfung wird entweder als einfache Standardprüfung oder als genauere Detailprüfung ausgeführt. Entscheidungskriterium hierfür ist der Auftragswert. Ist der Wert des Auftrags kleiner 5000 DM, wird die Standardprüfung durchgeführt, andernfalls die Detailprüfung.

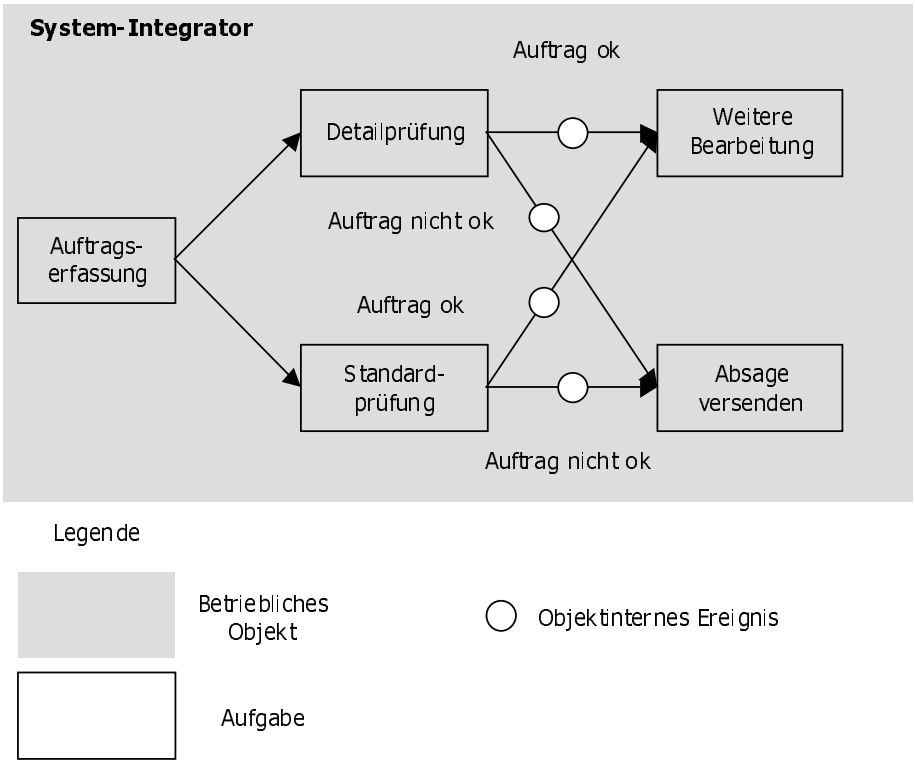


Abbildung 24: Auftragserfassung 1. Teil

Die weitere Bearbeitung wird auf die Teilnehmer des virtuellen Unternehmens verteilt, wie in Abbildung 25 in der SOM-Notation veranschaulicht ist. So muß von jedem Teilnehmer des virtuellen Unternehmens für seinen Aufgabenbereich die Prüfung der technischen Machbarkeit und Rentabilität durchgeführt werden. Jedes Unternehmen führt dann für sich die technische Prüfung und Rentabilitätsprüfung durch. Am Ende der separaten Prüfung steht die Entscheidung, ob der Auftrag bestätigt wird oder nicht. Diese Entscheidung fällt nicht automatisch, sondern im Rahmen einer Abstimmung. Dieses Auftauchen von Koordinationsverfahren mit gleichberechtigten Partnern, etwa Abstimmungen, ist typisch für virtuelle Unternehmen, da bei ihnen die Unternehmen autark bleiben, also keiner Hierarchie unterliegen.

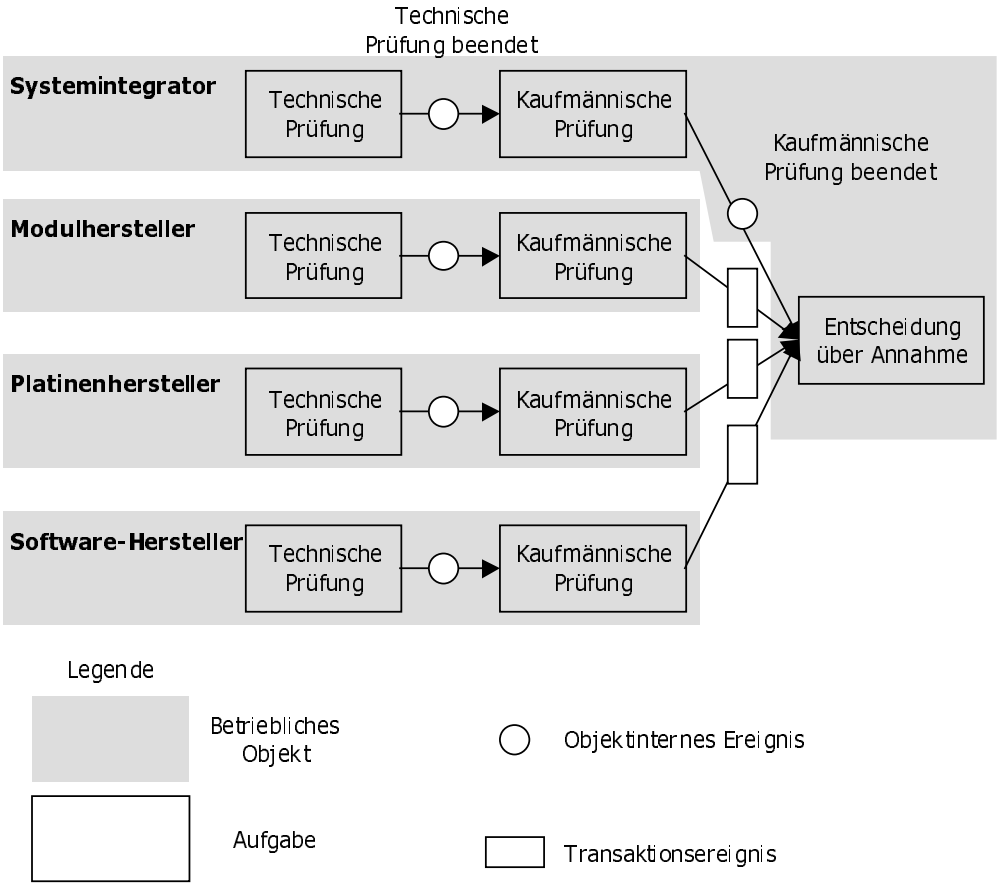


Abbildung 25: Auftragserfassung 2. Teil

3.2 Informationstechnische Anforderungen

In diesem Abschnitt wird ermittelt, welche informationstechnischen Anforderungen aus der umfassenden Unterstützung von weitreichenden Geschäftsprozessen resultieren. Dazu werden die Rahmenbedingungen für die informationstechnische Unterstützung weitreichender Prozesse mit Hilfe des oben beschriebenen Szenarios verdeutlicht, um dann die aus ihnen abzuleitenden informationstechnischen Anforderungen zu identifizieren.

3.2.1 Autarkie bei der informationstechnischen Umsetzung

Die Autarkie der Prozeßteilnehmer ist ein Faktor, der bei der Unterstützung weitreichender Workflows zunehmend [Riem98] Beachtung findet. Während bisher [Riem98] die Modellierung im Vordergrund stand, wird hier die Autarkie bei der Nutzung der informationstechnischen Ressourcen im Zentrum des Interesses stehen. Bei einem Prozeß in einer organisatorischen Einheit sind die Prozeßteilnehmer meist einer Hierarchie unterworfen, die sich beispielsweise in der Nutzung der gleichen informationstechnischen Ressourcen zur Prozeßunterstützung niederschlägt. So können beispielsweise durch unternehmensinterne Standardisierung alle Prozeßteilnehmer dazu gezwungen werden, ein bestimmtes WfMS zu nutzen. Bei weitreichenden Prozessen ist hingegen in hohem Maß mit der Autarkie der Prozeßteilnehmer zu rechnen, d.h. sie unterliegen keinen Weisungen.

Beispiel:
Die Teilnehmer des virtuellen Unternehmens möchten unterschiedliche Mittel zur

Umsetzung der kaufmännischen Prüfung einsetzen. Der Modulhersteller möchte ein selbstentwickeltes System verwenden, der Platinenhersteller sein R/3 System.

Die Prozeßteilnehmer müssen also die Umsetzung der informationstechnischen Prozeßunterstützung unabhängig von anderen wählen können. Es kann nicht davon ausgegangen werden, daß sich die Teilnehmer des weitreichenden Prozesses die Ressourcen zur informationstechnischen Unterstützung und deren Ausgestaltung vorgeben lassen. Dies ist allein wegen des Koordinationsaufwandes auch nicht realistisch. Aus dieser Betrachtung ergibt sich die Anforderung, daß die Mittel zur informationstechnischen Umsetzung der Prozesse nicht festgelegt sein dürfen.

Die Zuweisung informationstechnischer Ressourcen muß feingranular und zur Laufzeit möglich sein. So müssen die im Lauf des Workflows verwendeten Daten in unterschiedlichen Datenbanken ablegbar sein. Damit dies möglich ist, muß die Zuweisung von Mitteln zur informationstechnischen Umsetzung feingranular erfolgen können. Nicht verwendbar ist hingegen ein Granulat, bei dem beispielsweise nur eine Datenbank für die Speicherung aller Daten verwendet werden kann.

Die Zuweisung unterschiedlicher Ressourcen sowie der Wechsel derselben muß transparent für die Workflow-Schemata erfolgen können. Nur so ist beispielsweise erreichbar, daß ohne Beeinflussung der Workflow-Schemata ein Wechsel der Datenbank möglich ist.

Anforderung 1 **Autarkie der informationstechnischen Umsetzung**
Die informationstechnischen Ressourcen zur Umsetzung der weitreichenden Prozesse müssen dynamisch und feingranular zuweisbar sein.

Die Autarkie der Prozeßteilnehmer schließt ein, daß sie auch bereits bestehende Ressourcen für die informationstechnische Unterstützung des weitreichenden Prozesses verwenden.

Beispiel:
Im obigen Szenario wird angenommen, daß der Systemintegrator und der Softwarehersteller über eine leistungsfähige Personalverwaltung verfügen. Die anderen Unternehmen besitzen jedoch keine ausreichend leistungsfähige Personalverwaltung. Führt man ein WfMS mit einer leistungsfähigen Personalverwaltung ein, so können zwar die Bedürfnisse der zweiten Gruppe erfüllt werden, bei den Unternehmen mit einer leistungsfähigen Personalverwaltung kommt es jedoch zu einer Duplizierung von Funktionalität.

Diese Nutzung von bereits bestehenden Ressourcen ist nicht nur aus Gründen der Wiederverwendung und damit Wirtschaftlichkeit erstrebenswert, sondern aus Gründen der Konsistenzhaltung erforderlich. Ist das WfMS zur Nutzung von existierenden Ressourcen nicht in der Lage und dupliziert daher schon vorhandene Funktionalität, so wird damit auch eine Duplizierung von Daten notwendig. Wird diese Verwaltung nicht integriert, sondern neu implementiert, so müßten Informationen über die Mitarbeiter in das WfMS transferiert werden und dann vor allem in beiden Richtungen permanent konsistent gehalten werden. Änderungen an der Personalstruktur des Unternehmens müssen sowohl im WfMS, als auch in der bestehenden Personalverwaltung konfliktfrei eingebracht werden. Ein derartiger Abgleich ist aber oft schon deswegen nicht möglich, weil die beteiligten Systeme die entsprechenden Schnittstellen nicht offenlegen. Die Konsistenzhaltung ist dann nur organisatorisch, aber nicht mehr informationstechnisch möglich und es ist nur eine Frage der Zeit, bis widersprüchliche Daten entstehen.

Ein weiterer Grund für die Anforderung der Autarkie der informationstechnischen Umsetzung liegt in der Dynamik der Teilnehmermenge weitreichender Geschäftsprozesse, wie es beispielsweise in virtuellen Unternehmen der Fall ist. Sie erfordert davon auszugehen, daß wechselnde Partner zusammenarbeiten. Diese können sogar in einem Konkurrenzverhältnis zu den bisherigen Partnern stehen. Daher liegt es in hohem Maße im Eigeninteresse der Prozeßteilnehmer, die Kontrolle über die informationstechnische Umsetzung zu behalten.

> Beispiel:
> Nach Abschluß des oben beschriebenen virtuellen Unternehmens kommt es zur Bildung eines neuen virtuellen Unternehmens, jedoch mit einem anderen Platinenhersteller (B) wie in Abbildung 26 dargestellt. Der ursprüngliche Platinenhersteller (A) nimmt an einem anderen virtuellen Unternehmen teil, das zu dem neugebildeten virtuellen Unternehmen in Wettbewerb steht.

Die Konsequenzen wären fatal, wenn der ursprüngliche Platinenhersteller A das WfMS verwaltet, das von den anderen Teilnehmern benötigt wird. Bei seinem Ausscheiden würden die verbleibenden Unternehmen ihrer Workflow-Unterstützung beraubt. Diese Problematik betrifft nicht nur das WfMS, sondern auch die Workflow-Schemata. Workflow-Schemata repräsentieren für die Unternehmen in hohem Maße wichtiges Prozeßwissen, das sicherlich nicht in die Hände der Konkurrenz gelangen sollte. Sie werden sich daher auf kein WfMS einlassen, bei dem sie damit rechnen müssen, daß dies geschieht, wie beispielsweise bei einem zentralisierten WfMS. Jeder Prozeßteilnehmer muß also die Kontrolle über die von ihm zur Prozeßunterstützung verwendeten informationstechnischen Ressourcen behalten.

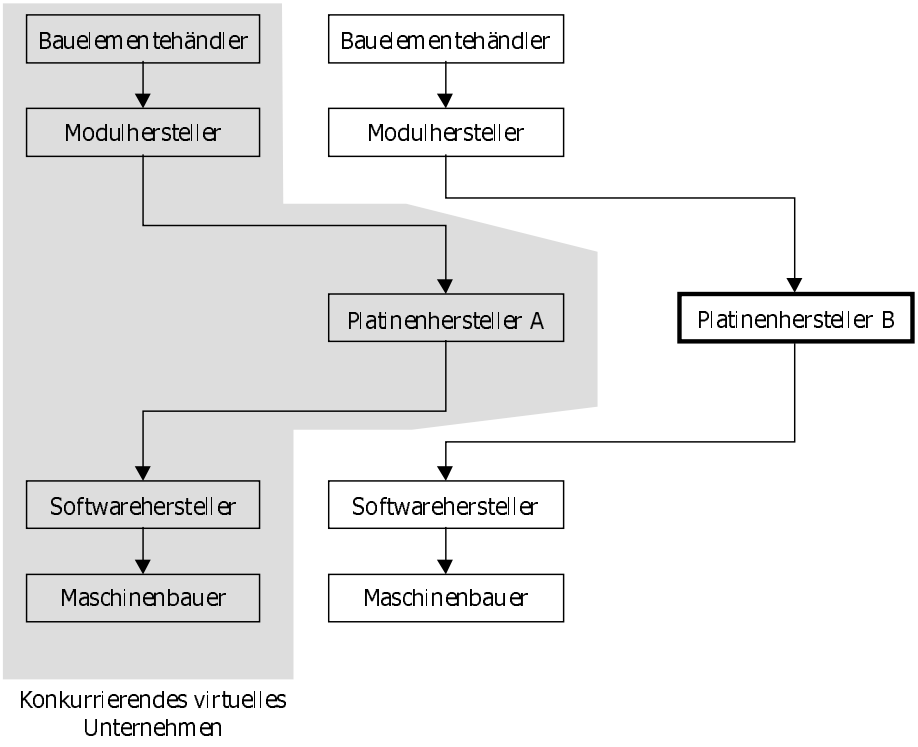


Abbildung 26: Wechselnde Geschäftspartner in einem virtuellen Unternehmen

3.2.2 Inkrementelle Abbildung von Prozeßschemata auf Workflow-Schemata

Durch sich immer schneller verändernde Marktbedingungen sind die Unternehmen gezwungen, immer häufiger ihre Prozesse anzupassen. Aber nicht nur die Anzahl der Prozeßänderungen ist höher, auch die Qualität der Prozeßänderungen hat sich verändert. Sie sind tiefgreifender als

bisher. Beide Anforderungen treffen besonders für weitreichende Prozesse zu. Bei ihnen existieren durch die Vielzahl und Heterogenität der Teilnehmer deutlich mehr Quellen für Änderungen an den Prozessen als bei Prozessen, die auf Abteilungsebene beschränkt sind. Auch die zeitliche Begrenztheit von virtuellen Unternehmen erfordert, die zu ihrer Bildung notwendigen Anpassungen schnell durchführen zu können. Eine Prozeßunterstützung, die zur Umsetzung von Prozeßänderungen länger braucht als die Lebenszeit des virtuellen Unternehmens, ist sicherlich kaum brauchbar.

Die Veränderung eines Prozeßschemas wird zunächst an Hand des Szenarios motiviert.

> Beispiel:
> Das virtuelle Unternehmen des Szenarios floriert und gewinnt zunehmend auch an internationaler Kundschaft. Deshalb müssen Aufträge darauf überprüft werden, ob sie von einem Auftraggeber aus einem Embargoland stammen. Daher ist eine Anpassung des Prozesses Angebotserstellung notwendig. So soll nach der detaillierten Eingangsprüfung oder Standardprüfung ein zusätzlicher Arbeitsschritt "Embargoprüfung" eingeführt werden. Falls der potentielle Auftraggeber aus einem Embargoland stammt, soll eine dementsprechende Absage verschickt werden. Andernfalls soll die Bearbeitung wie bisher erfolgen.

In Abbildung 27 ist dieser erweiterte Prozeß als Vorgangs-Ereignis-Schema der SOM-Notation dargestellt.

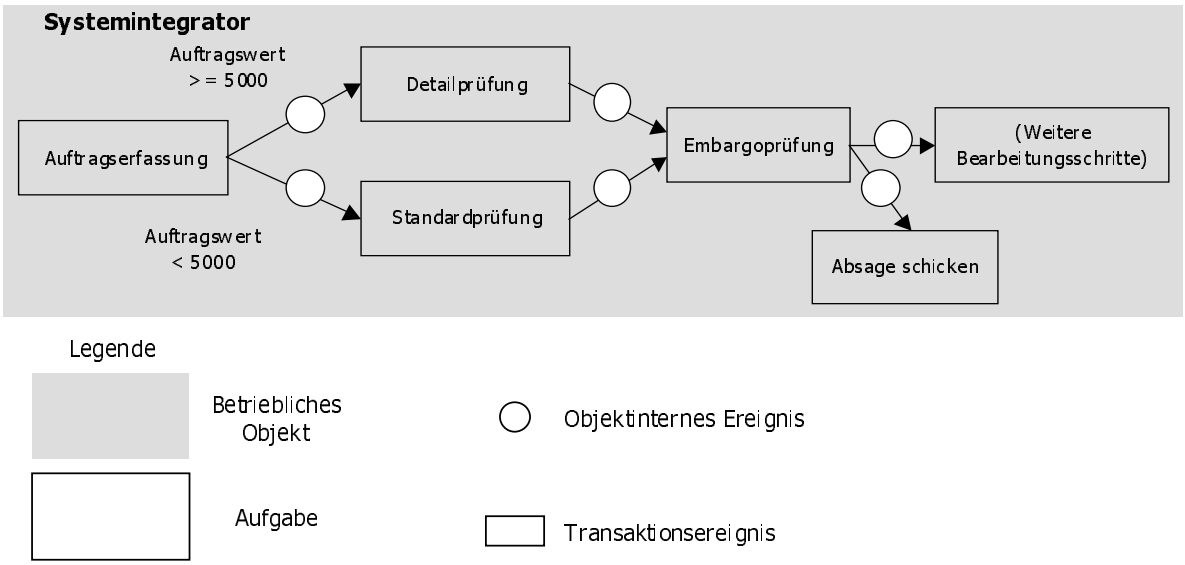


Abbildung 27: Vorgangs-Ereignis-Schema des Beispielprozesses

Der erste Schritt zur Umsetzung dieses veränderten Prozeßschemas ist die Abbildung des Prozeßschemas auf ein Workflow-Schema. Bei kleinen Prozeßschemata kann dazu eine Gesamtabbildung des Prozesses erfolgen. Bei der umfassenden Unterstützung weitreichender Prozesse ist jedoch mit deutlich größeren Schemata rechnen. Ein Grund hierfür ist die weitere Ausdehnung oder aber die Einbeziehung mehrerer Organisationen wie Abbildung 23 exemplarisch zeigt: Die technische Prüfung und die Rentabilitätsprüfung sind auf mehrere Teilnehmer des virtuellen Unternehmens verteilt.

Hinzu kommt, daß weitreichende Prozesse verstärkt miteinander in Beziehung stehen. Die Totalabbildung der Prozeß- auf die Workflow-Schemata ist bei derartigen umfangreichen und miteinander verknüpften Schemata nicht mehr anwendbar, da eine Vielzahl sich fortpflanzender

Änderungen erzeugt wird. Als Folge ist ein auch schon bei isolierten WfMS zu beobachtender Effekt zu erwarten, nämlich das Auseinanderdriften von (Geschäfts-)Prozeßschemata und Workflow-Schemata. So wird zwar die Abbildung einmal durchgeführt, nachfolgende Änderungen werden aber wegen des hohen Aufwands nicht mehr im Prozeßschema nachgeführt, statt dessen wird das Workflow-Schema direkt geändert. Das Workflow-Schema enthält also Änderungen der Realprozesse, die nicht mehr im Geschäftsprozeßschema repräsentiert sind. Dies hebt nicht nur die Unterscheidung (im Sinne von Abschn. 1.1) von fachlichem und informationstechnischem Modell auf, sondern erschwert bei weitreichenden Prozessen die Kommunikation erheblich. Die Teilnehmer eines Prozesses müssen sich dann auf Grund eines Workflow-Schemas und nicht mehr auf Basis eines leichter verständlichen (Geschäfts-)Prozeßschemas verständigen.

Gefordert ist daher eine synchrone Fortschreibung beider Schemata. Damit dies mit vertretbarem Aufwand geschehen kann, sollte eine Strategie der „kleinen Schritte“ verfolgt werden. Daher ergibt sich die Anforderung die Abbildung von Geschäftsprozessen auf Workflow-Schemata inkrementell durchzuführen.

Anforderung 2 **Inkrementelle Abbildung der Prozeß- auf Workflow-Schemata**
Die Unterstützung weitreichender Prozesse erfordert es, Änderungen der Prozeßschemata mit minimalem Aufwand auf die Workflow-Schemata zu übertragen.

3.2.3 Erweiterbarkeit des Workflow-Modells

Durch die inkrementelle Abbildung des veränderten Geschäftsprozeßschemas wird ein verändertes Workflow-Schema geschaffen. Damit dies vom WfMS verarbeitet werden kann, darf es keine dem WfMS unbekannten Schemaelemente enthalten. Bei der Unterstützung weitreichender Prozesse ist aber damit zu rechnen, daß Schemata Elemente enthalten, die vom WfMS nicht unterstützt werden.

> Beispiel:
> Die Embargoprüfung ist bisher nicht Bestandteil des Workflow-Modells des WfMS. Daher kann ein Schema, das das Element Embargoprüfung enthält, nicht verarbeitet werden. Es ist notwendig, das Workflow-Modell des WfMS um das Element Embargoprüfung zu erweitern.

Ein weiteres Beispiel sind Modellelemente zur Unterstützung von gleichberechtigten Koordinationen. Firmenübergreifende Prozesse spielen sich häufig zwischen gleichberechtigten Partnern ab, so daß nicht mehr hierarchische Koordinierungselemente, sondern Abstimmungen usw. benötigt werden. Hierfür sind eine Vielzahl unterschiedlicher Verfahren denkbar, die nicht im voraus bestimmbar sind. Daher kann auch das Workflow-Modell eines WfMS diese nicht in ihrer Gesamtheit erfassen. Auch aus diesem Grund muß das Workflow-Modell bei der Unterstützung weitreichender Prozesse erweiterbar sein.

Zur Umsetzung von Prozeßänderungen mit neuen Elementen muß es möglich sein, die Menge der vom WfMS unterstützten Workflow-Schemaelemente, d.h. das Workflow-Modell des WfMS, zur Laufzeit des WfMS zu erweitern. Veranschaulicht ist dies in Abbildung 28. Die Mindestforderung für die Erfüllung der Erweiterbarkeit ist, daß zusätzliche Workflow-Modellelemente in der Entwurfsphase, aber vor der Bildung eines Schemas eingebracht werden können. Vorteilhafter ist es hingegen, die Erweiterung zu einem Zeitpunkt einbringen zu können, an dem bereits Schemata existieren (Schemata A und B zum Zeitpunkt t1). Neu gebildete Schemata wie (C) können dieses zusätzliche Modellelement nutzen. Der Idealzustand ist

schließlich, wenn neue Modellelemente auch für gerade im Entwurf befindliche Schemata verwendet werden können, hier also für die Schemata A und B.

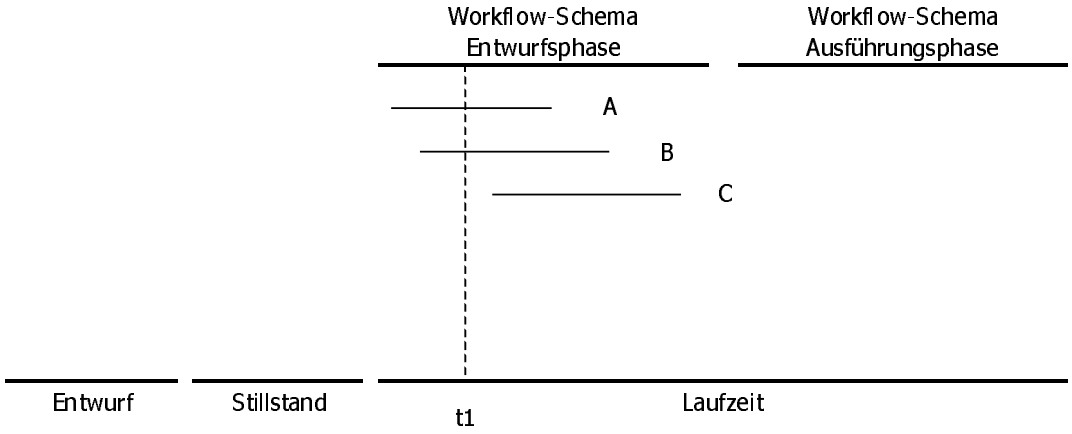


Abbildung 28: Zeitpunkte für das Einbringen von Modellerweiterungen

Aus diesen Überlegungen ergibt sich die dritte Anforderung:

Anforderung 3 **Erweiterbarkeit**
Die informationstechnische Unterstützung weitreichender Prozesse erfordert es, Erweiterungen des zu Grunde liegenden Workflow-Modells zur Laufzeit umsetzen können.

Die Zuweisung von Ressourcen und die oben aufgestellte Forderung der Erweiterbarkeit sind strikt voneinander zu trennen. So kann ein WfMS keinerlei Erweiterungen bezüglich seines Workflow-Modells zulassen, aber zur Umsetzung der Elemente des Workflow-Modells die Nutzung beliebiger externer Ressourcen zulassen. Umgekehrt ist auch Erweiterbarkeit ohne die Möglichkeit zur wahlfreien Nutzung von Ressourcen möglich. In diesem Fall kann das Workflow-Modell leicht erweitert werden, aber bezüglich der verwendeten Ressourcen besteht keinerlei Wahlfreiheit.

3.2.4 Flexibilität

Im nächsten Schritt muß dieses veränderte Workflow-Schema in das WfMS eingebracht werden. D.h. es muß an Stelle des bisherigen Schemas die Grundlage der Erzeugung von Workflow-Ausprägungen bilden.

Beispiel:
Es soll angenommen werden, daß das Geschäftsprozeßschema aus Abbildung 27 auf ein Workflow-Schema abgebildet wurde, wie in Abbildung 29 dargestellt. Dieses veränderte Schema muß in das WfMS eingebracht werden. Dies bedeutet, daß zumindest neu gebildete Ausprägungen des Workflows auf diesem veränderten Schema beruhen.

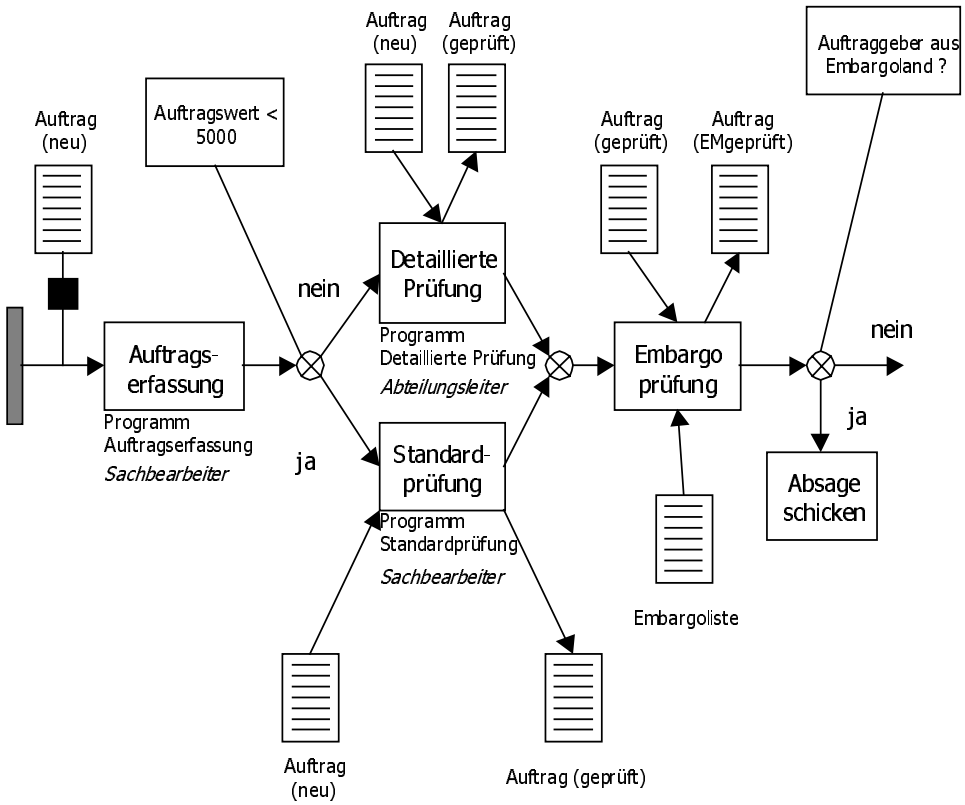


Abbildung 29: Darstellung des veränderten Beispielworkflows in EventFlow$_L$

Wegen der angesprochenen Häufigkeit von Prozeßänderungen und dementsprechenden Veränderungen der Workflow-Schemata ist es notwendig, die Änderungen von Workflow-Schemata zur Laufzeit durch das WfMS umsetzen zu können. Was dies genau bedeutet, wird mit Hilfe von Abbildung 30 veranschaulicht.

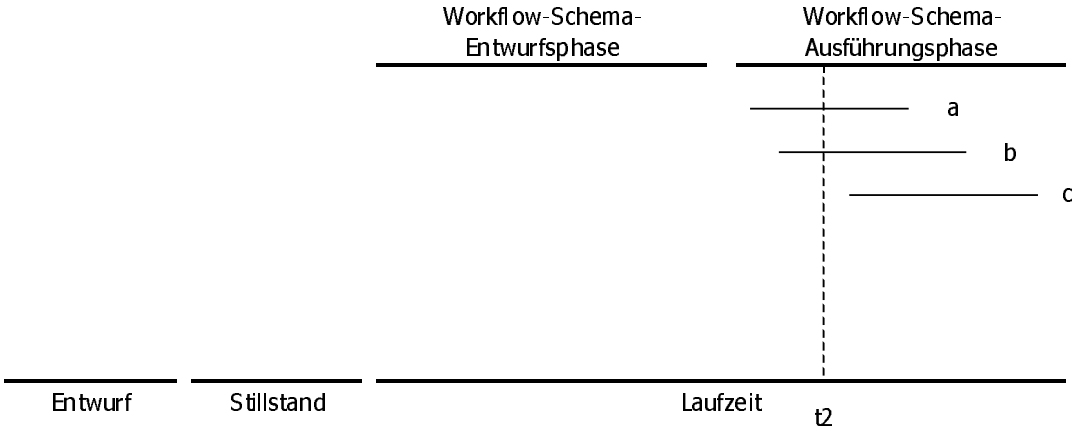


Abbildung 30: Zeitpunkte für das Einbringen von Schemaänderungen

Die Mindestforderung für die Erfüllung der Flexibilität ist, daß veränderte Workflow-Schemata in der Workflow-Schema-Entwurfsphase, aber noch ohne laufende Workflow-Ausprägungen, eingebracht werden können. Dies bedeutet, daß alle noch laufenden Workflow-Ausprägungen beendet sein müssen. Vorteilhafter ist es hingegen, die Änderung zu einem Zeitpunkt einbringen zu können, an dem noch laufende Workflow-Ausprägungen existieren (Ausprägungen a und b zum Zeitpunkt t2). Diese werden jedoch von der Änderungen nicht berührt. Neu gebildete Ausprägungen (Ausprägung c) werden auf der Basis des neuen Schemas gebildet.

Anforderung 4 **Flexibilität**
Die informationstechnische Unterstützung weitreichender Prozesse muß Veränderungen von Workflow-Schemata zur Laufzeit des WfMS umsetzen können.

Der Begriff der Flexibilität ist von dem der Ausnahmebehandlung [CCPP96a], [CCPP98] strikt zu trennen. Bei der Ausnahmebehandlung gilt es, Abweichungen vom durch das Workflow-Schema vorgegebenen Verlauf einer Workflow-Ausprägung zu unterstützen [Sheth97], [DaRe98], [ReDa98], [Sieb99]. Ein Beispiel wäre die Veränderung des Ablaufs von Ausprägung b zum Zeitpunkt t1, ohne daß die Ausprägung a oder die Ausprägung c betroffen ist. Das Vorhandensein einer Ausnahmebehandlung bezeichnet also die Möglichkeit, ad-hoc Änderungen einer Workflow-Ausprägung durchzuführen, von der nur diese beeinflußt wird.

3.2.5 Skalierbarkeit

Eine weitere Rahmenbedingung bei der Unterstützung weitreichender Prozesse ist die Notwendigkeit, eine große Zahl von Prozessen zu unterstützen, also eine umfassende Unterstützung zu bieten. D.h. es gilt nicht mehr nur eine beschränkte Zahl von Prozessen innerhalb einer Abteilung, sondern die Gesamtheit der Prozesse eines Unternehmens zur Ausführung zu bringen. Dies bedeutet, daß die Workflow-Unterstützung eine ausreichende Leistungsfähigkeit besitzen muß. Voraussetzung hierfür ist die Skalierbarkeit der informationstechnischen Systeme zur Workflow-Unterstützung.

Anforderung 5 **Skalierbarkeit**
Die Unterstützung weitreichender Prozesse erfordert eine skalierbare Workflow Unterstützung.

3.3 Zusammenfassung

An Hand dieses Szenarios aus dem VORREITER-Projekt konnte die Relevanz der in der Einleitung identifizierten Rahmenbedingungen für die Unterstützung weitreichender Prozesse aufgezeigt werden. In einem weiteren Schritt wurden informationstechnische Anforderungen entwickelt. Es sind dies die:

- Autarkie bei der informationstechnischen Umsetzung
- Flexibilität
- Erweiterbarkeit des Workflow-Modells
- Skalierbarkeit
- Inkrementelle Abbildung von Prozeßschemata auf Workflow-Schemata

Die ersten vier dieser Anforderungen werden in den nächsten Kapiteln behandelt, die inkrementelle Abbildung von Prozeßschemata auf Workflow-Schemata erst in Kapitel 11.

Es soll nicht verschwiegen werden, daß mit der Unterstützung weitreichender Prozesse noch eine ganze Frage zusätzlicher Anforderungen auftauchen. So werden mit hoher Wahrscheinlichkeit öffentliche Netze als Kommunikationsinfrastruktur genutzt. Daher ist es notwendig, für eine geeignete Verschlüsselung der Informationen zu sorgen, die im Rahmen eines Prozesses ausgetauscht werden. Ebenfalls ist es notwendig sicherzustellen, daß zwar die Prozeßteilnehmer aber nicht Unberechtigte an der Prozeßausführung teilnehmen. Daher müssen auch dementsprechende Authentifizierungsmechanismen bereitstehen. Die Berücksichtigung

dieser und weiterer Anforderungen würde jedoch den Rahmen dieser Arbeit sprengen. Sie sollen hier nicht weiter verfolgt werden, zumal es umfangreiche Forschungsaktivitäten gibt, die sich speziell mit diesen Fragestellungen beschäftigen [HePe97].

4 Stand der Forschung

In diesem Kapitel wird untersucht, ob Forschungsarbeiten die im vorangegangenen Kapitel identifizierten informationstechnischen Anforderungen erfüllen, also in wieweit sie die Autarkie bei der informationstechnischen Umsetzung, Erweiterbarkeit, Flexibilität und Skalierbarkeit bieten. Die Ansätze sind danach gruppiert, ob sie Standardisierungsansätze sind, eine oder mehrere der obigen Anforderungen adressieren oder die Unterstützung weitreichender Prozesse als direktes Ziel haben. Danach folgt eine Analyse, die klärt, wieso die existierenden Ansätze nicht in der Lage sind, die gestellten Anforderungen vollständig zu erfüllen. Es werden aber genauso auch verwendbare Lösungsbeiträge identifiziert.

4.1 Standardisierungsansätze

Ziel des WfMC-Standards [WfMC] ist die Bereitstellung einer standardisierten Architektur für WfMS. Die meisten kommerziellen WfMS ähneln zumindest grob dem durch die WfMC-Architektur vorgegebenen Referenzmodell. Dieses Referenzmodell besteht aus fünf Funktionsblöcken und Schnittstellen zwischen ihnen. Die Funktionsblöcke des WfMC-Referenzmodells sind:

- **Process Definition/Builder Tools**:
  Spezifikations- und Entwurfswerkzeuge für Workflows

- **Workflow Enactment Service (WES)**:

  Aufgabe des Workflow Enactment Service ist die Bildung von Ausprägungen, Ausführung und Überwachung von Workflows, die Zuordnung der Aufgaben zu Organisationseinheiten, die Verwaltung von persönlichen Aufgabenlisten (Worklists), die Aktivierung von Anwendungen sowie die Verwaltung der am Workflow teilnehmenden Mitarbeiter und ihrer Rollen. Die proaktive Ausführung ist Aufgabe der Workflow-Maschine (= Workflow-Engine).

- **Workflow Client Applications**:
  Sie stellen die Benutzungsschnittstelle zum am Workflow teilnehmenden Benutzer bereit.

- **Invoked Applications**:
  Anwendungen außerhalb des WfMS, die zur Durchführung des Workflows benötigt werden

- **Administration & Monitoring Tools**:
  Werkzeuge zur Verwaltung und Überwachung laufender Workflows

Eine Übersicht der Bestandteile des WfMC-Referenzmodells gibt Abbildung 31:

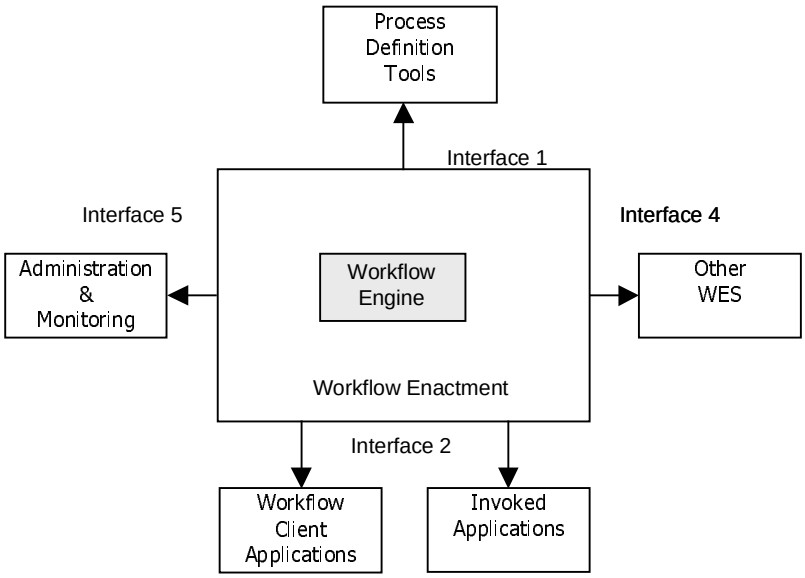


Abbildung 31: WfMC-Referenzmodell

Zwischen der Workflow-Maschine und weiteren Werkzeugen waren ursprünglich 5 Schnittstellen vorgesehen. Die dritte Schnittstelle (Interface 3) wurde jedoch in die zweite Schnittstelle überführt.

- **Schnittstelle 1**
  Sie legt ein allgemeines Austauschformat zwischen dem Process Definition Tools und der Workflow-Engine fest.

- **Schnittstelle 2**
  In dieser Schnittstelle ist die Kommunikation mit den Benutzerschnittstellen und externen Anwendungen spezifiziert.

- **Schnittstelle 4**
  Diese Schnittstelle soll die Interoperabilität zwischen verschiedenen Implementierungen des Referenzmodells sicherstellen.

- **Schnittstelle 5**
  Werkzeuge zur Verwaltung und Überwachung können den Zustand der Workflow-Maschine über diese Schnittstelle abfragen.

WfMS nach dem WfMC-Standard können Schemaänderungen meist flexibel umsetzen. Dennoch ist der WfMC-Standard Ziel kritischer Anmerkungen einer Vielzahl von Arbeiten wie beispielsweise [CGPP97], [Moha96], [SBMW96], [DKMS96], [WWWK96]. Grund ist das zentralisierte Verarbeitungsmodell, das eine ständige Interaktion mit der zentralen Workflow-Maschine zur Steuerung der Kontroll- und Datenflüsse erfordert. Dies macht die Workflow-Maschine zu einem Leistungs- und Zuverlässigkeitsengpaß. Daran ändert auch die in Schnittstelle 4 vorgesehene Zusammenarbeit mit anderen WfMS nichts, da durch sie nur eine Partitionierung, aber keine Verteilung erreicht werden kann. Selbst von den über die Schnittstelle 2 angesprochenen externen Anwendungen zur Unterstützung von Workflow-Operationen wird

verlangt, daß sie sich auf dem gleichen Rechner wie das WfMS befinden. Beides schränkt die Skalierbarkeit dieses Ansatzes drastisch ein [AAAM97] [GeHS95]. Der WfMC Ansatz erreicht nur eine sehr grobgranulare Zuweisung informationstechnischer Ressourcen, da keine weitere Zerlegung der Funktionalität der Workflow-Maschine erfolgt. Die Workflow-Maschine kann daher nur in ihrer Gesamtheit ausgetauscht werden. Auch eine Erweiterung des Workflow-Modells zur Laufzeit ist nicht möglich.

Die grundsätzlichen Probleme des WfMC-Standards werden auch durch eine Anbindung an das WWW [WWW] nicht behoben. Aus diesem Grund gelten die obigen Argumente auch für Ansätze, die eine Erweiterung des WfMC-Ansatzes um eine WWW Anbindung durchführen. Beispiele hierfür finden sich in [BHJS96], [EdGL96], [LKSS96], [Wewe96], [EnWa98], [MCPM98].

Weitere Standardisierungsansätze stellen die Workflow-Facility und die Business Object-Facility des CORBA Standards dar [OMGW]. An Beginn ihrer Entwicklung stand die deutliche Kritik am WfMC-Modell [PaPC97], [Schu97d], [SBMW96]. Diese beeinflußte die Entwicklung jedoch nur unwesentlich, weswegen die für den WfMC-Ansatz vorgebrachten Einschränkungen auch für den jetzt entwickelten Ansatz [WFFA] gelten.

4.2 Ansätze, die einzelne Anforderungen aus der Unterstützung weitreichender Prozesse adressieren

4.2.1 Broker/Services Modell und EVE

Die Kombination aus Broker/Services Modell in Verbindung mit der ereignisbasierten Middleware EVE („event engine“) [GeTo97], [GeTo98], [ToGD96], [ToGD97] [GeTD98] wird zur Unterstützung von verteilten Workflows sowie zur Integration von Umgebungsdiensten für den Operations- und Organisationsaspekt eingesetzt. Im Zentrum des Interesses steht die Absicht, die Verwendbarkeit von EVE und dem Broker/Services Modell für die Workflow-Unterstützung aufzuzeigen. Dies wird dadurch unterstrichen, daß es keinen eigenen Namen für die Workflow-Unterstützung gibt. Statt dessen ist immer von Workflow-Unterstützung auf der Basis von EVE und dem Broker/Services Modell die Rede.

Workflows werden in diesem Ansatz als Regeln repräsentiert, die von EVE verarbeitet werden. Auf diese Weise steuert EVE das Zusammenspiel von Brokern, die die am Workflow teilnehmenden Personen und Softwaresysteme repräsentieren. EVE ist in sogenannte Knoten organisiert, die jeweils einen Server besitzen. EVE kann verteilte, zusammengesetzte Ereignisse erkennen und Ereignisse weiterleiten.

In einem Wörterbuch[4] werden die Informationen über Broker, Ereignistyp und ECA-Regeln abgelegt. Zur späteren Analyse und Auswertung können Ereignishistorien persistent gespeichert werden. Mechanismen für die Unterstützung von Ausnahmeereignissen ermöglichen die Behandlung von Workflow-Ausnahmefällen. Broker werden über ihre statische Beschreibung (state), ihre Interaktion mit der Umwelt sowie ihr Verhalten charakterisiert. Die statische Beschreibung eines Brokers enthält die von ihm genutzten Schnittstellen sowie die Beschreibung der Datenobjekte, die er zur Ausführung seiner Aufgaben benötigt. Die Interaktion mit anderen Brokern wird über die von ihm ausgesandten und entdeckten Ereignisse definiert. Die versandten Ereignisse stellen Dienstanforderungen dar, während die empfangenen Ereignisse die Ergebnisse von Dienstanforderungen darstellen.

[4] Repository wurde durch Wörterbuch ersetzt

Das Verhalten eines Brokers wird über ECA-Regeln definiert. So wird einerseits mit Regeln festgelegt, wie ein Broker auf eine Dienstanforderung reagiert, andererseits wird auch das Verhalten des Brokers auf die Ergebnisse von Dienstausführungen festgelegt. Die Ereignisse, die in den ECA-Regeln spezifiziert werden, können einfach oder zusammengesetzt sein. Zusammengesetzte Ereignisse werden über Konstruktoren wie Sequenz, Disjunktion usw. aus einfachen und zusammengesetzten Ereignissen zusammengesetzt.

Die Broker werden in drei Gruppen eingeteilt. Externe Broker versenden nur Ereignisse, bieten jedoch keine für andere Broker verwendbaren Dienste an. Interne Broker repräsentieren die am Workflow teilnehmenden Personen oder Softwaresysteme und werden wiederum unterteilt in Schnittstellenbroker, Agenda-Broker und Wrapper-Broker. Schnittstellenbroker dienen der Kapselung von Software-Systemen, deren Funktionalität über eine Schnittstelle zugänglich ist. Zu ihnen gehören auch die Systembroker, deren Implementierung Bestandteil des Workflow-unterstützenden Systems ist. Wrapper-Broker kapseln externe Software-Systeme. Agenda-Broker stellen die Schnittstelle zu Personen dar, die am Workflow teilnehmen. Gruppenbroker, als dritte Gruppe, dienen der Darstellung des organisatorischen Aufbaus. Mitglieder einer Gruppe können gleichzeitig auch zu anderen Gruppen gehören. Gruppenbroker besitzen kein Verhalten.

Die Ausführung von Workflows beginnt, indem ein Broker ein Ereignis an den lokalen EVE-Server schickt. Dieses wird ausgewertet und die anwendbaren Regeln im EVE-Wörterbuch werden aktiviert. Diese erzeugen ihrerseits Ereignisse, die Broker auf dem gleichen oder anderen Server aktivieren. Diese Broker übernehmen dann die weiteren Schritte des Workflows und senden ihrerseits Ereignisse aus, die die Ausführung des Workflow fortführen.

Die Flexibilität des EVE/Broker-Services Ansatzes ist ähnlich der von traditionellen WfMS, wenn man die Regelsprache als Workflow-Sprache interpretiert, mit deren Hilfe das Workflow-Schema gebildet wird. Schemaänderungen können durch Regelanpassungen flexibel wiedergegeben werden. Allerdings resultieren aus der Verwendung von Regeln Schwächen, die auch von den Autoren angesprochen werden. So wird in einer weiteren Veröffentlichung [GeTo97] auf die mangelnde Leistungsfähigkeit verwiesen, die bereits bei wenigen hundert Regeln Probleme aufwirft. Der EVE/Broker-Services Ansatzes bietet keine Möglichkeit für die Erweiterung um zusätzliche Aspekte. Die Autarkie ist für Elemente des Operations- und Organisationsaspekts gewährleistet, da über das Broker-Services Modell Ressourcen für diese Aspekte zugewiesen werden können. Für andere Aspekte ist dies jedoch nicht möglich. So ist EVE elementarer Bestandteil des Lösungskonzepts, der nicht ersetzt werden kann.

Der EVE /Broker-Services Ansatz versucht das Problem der verteilten Workflow-Ausführung durch die Erkennung und Verarbeitung verteilter Ereignisse zu lösen. Dabei wird aber übersehen, daß diese Erkennung und Verarbeitung selbst nicht verteilt arbeitet. Sie enthält zentralisierte Elemente wie die Ereignishistorie und ein Wörterbuch, in dem für die Ausführung von Workflow-Ausprägungen notwendige Informationen abgelegt sind [GeTo98]. Hierzu gehören Workflow-Typ, Spezifikation der verwendeten Broker einschließlich ihres Verhaltens und ihrer Verantwortlichkeiten, organisatorische Beziehungen, Ereignistypen und ECA. Auf diese Weise kann keine volle Verteilung der Workflow-Ausführung erreicht werden, da der Kontrollfluß immer zu den EVE Servern zurückkehren muß. Es ist allenfalls eine Partitionierung von Workflows möglich. EVE erkennt zwar verteilte Ereignisse, führt diese Erkennung aber zentralisiert durch.

4.2.2 Exotica/FMQM

Der Exotica/FMQM Ansatz [AAEM95], [AGMK95], [MAAA95a], [MAGK95b], [MAGK95c], [KAGM96] hat die Verbesserung der Zuverlässigkeit von WfMS zum Ziel. Dabei geht der Exo-

tica/FMQM Ansatz einen sehr speziellen Weg bei der Workflow-Unterstützung, indem er persistente Nachrichtenwarteschlagen zur Workflow-Unterstützung verwendet. Kernidee ist dabei, auf eine zentralisierte Datenbank zur Speicherung der Workflow-Informationen zu verzichten und so zu einer höheren Ausfallsicherheit des Gesamtsystems zu gelangen. Durch Verwendung der persistenten Nachrichtenwarteschlangen können selbst beim Ausfall eines Teilsystems andere Teilsysteme die Arbeit fortsetzen.

Die Flexibilität des Exotica/FMQM Ansatzes ist relativ gering, da Workflow-Schemaänderungen nur durch Neuübersetzung eingebracht werden können. Allerdings erreicht er eine hohe Skalierbarkeit durch eine ausprägungsverteilte Ausführung der Workflows. Zur Durchführung von Erweiterungen ist eine Neuübersetzung der Knotenverwalter notwendig ist. Die flexible Zuweisung informationstechnischer Ressourcen und deren transparente Nutzung spielten bei der Konzeption von Exotica/FMQM nur eine untergeordnete Rolle. Autarkie besteht nur durch die Möglichkeit, externe Anwendungen für Elemente des Operationsaspektes zu verwenden. Für andere Aspekte können keine Ressourcen zugewiesen werden.

4.2.3 INCOME/WF

Der INCOME/WF Ansatz [ObSS94], [ObSZ96], [OSSW97] hat die Unterstützung für schwach und unstrukturierte Abläufe, sowie die leichte Anpaßbarkeit der Workflow-Schemata zum Ziel. Eine zentrale Rolle nimmt dabei die Modellierung von Workflows durch höherstufige Petri-Netze in Form der „nested relation/transition net“ (NR/T nets) [Ober96a] ein, die als Grundlage für die INCOME/WF Workflow-Modellierungssprache dient [Ober96b]. Diese werden durch eine Workflow-Maschine interpretiert und ausgeführt und es wird eine hohe Flexibilität erreicht. Dazu greift die Workflow-Maschine auf das Workflow-Schema-Wörterbuch, Workflow-Dictionary genannt, zu. Es enthält die durch NR/T Netze modellierten Workflows, die Beschreibung der Daten, die in den Workflows verarbeitet werden, sowie die Statusinformationen der gerade ausgeführten Workflows. Erweiterbarkeit war kein vorrangiges Ziel von INCOME/WF. Die Autarkie bezüglich der informationstechnischen Umsetzung ist nur für den Operationsaspekt vorhanden. Durch die Verwendung von Python[Pyth] können die Bestandteile von INCOME/WF transparent verteilt werden und so Schemapartitionen ausgeführt werden.

4.2.4 MetuFlow

Der MetuFlow Ansatz hat die Unterstützung verteilter Workflows zum Ziel und stützt sich auf die Arbeiten von [Sing96a], [Sing96b]. Der MetuFlow-Ansatz selbst wird in [GACT97] beschrieben. Die Erweiterung im MARIFlow Projekt ist in [DASB98], [MARI] dargestellt. Es handelt sich dabei um einen Ansatz, der auf der Überwachung von Ereignisabhängigkeiten beruht. Workflows werden also in eine Menge von Ereignisabhängigkeiten übersetzt. Dabei werden die Abhängigkeiten zwischen verschiedenen Teil-Workflows durch Ereignisabhängigkeiten wiedergegeben. Diese werden durch sogenannte Guards überwacht. Guards sind temporale Ausdrücke, die das Auftreten von Ereignissen nur zulassen, wenn die in ihnen festgelegten Bedingungen erfüllt sind. Aktoren verwalten den Guard eines Ereignisses und führen die Kommunikation mit anderen Aktoren durch. Mit ihnen werden die Abhängigkeiten zwischen den einzelnen Teil-Workflows wiedergegeben.

Die Flexibilität des MetuFlow Ansatzes ist wegen der übersetzenden Arbeitsweise und der Notwendigkeit, dementsprechende Guards zu erzeugen, gering einzuschätzen. MetuFlow ist außerdem nur statisch erweiterbar. Die zur informationstechnischen Umsetzung verwendeten Ressourcen können nur für den Operationsaspekt feingranular und zur Laufzeit gewählt werden, wobei CORBA die transparente Nutzung erlaubt. Um eine verteilte Verarbeitung zu ermöglichen, wird auf einen zentralen Mechanismus verzichtet, statt dessen werden Aktoren

[Agha86] und Guards eingesetzt. Auf diese Weise wird ein schemaverteiltes Ausführungsmodell erreicht. Problematisch dabei ist allerdings, daß es bei der Erzeugung der Guards zu einer kombinatorischen Explosion kommt [ASSR93], weswegen der Ansatz nicht skalierbar ist.

4.2.5 Mobile

Mobile [Jabl94], [SJKB94], [BuJa95], [BuJa96], [JaBS97] hat die modulare, und insbesondere erweiterbare Gestaltung von Workflow-Management-Systemen zum Ziel. Es ist einer der ersten Ansätze, der sich explizit mit der vorteilhaften Gestaltung von WfMS-Architekturen beschäftigt. Eine Verfeinerung des Ansatzes findet sich in [Schu98]. Grundlage für seine Entwicklung ist das in Abschnitt 2.3.1 beschriebene Workflow-Metamodell. Auf seiner Basis werden sowohl eine modulare Workflow-Sprache, als auch eine modulare Architektur für Workflow-Management-Systeme entwickelt. Die Mobile-Architektur sieht die Schaffung eines Workflow-Systems durch Module vor, die jeweils einen Aspekt in seiner Gesamtheit unterstützen, jedoch keine weitere Zerlegung durchführen. Mobile verwendet keine objektorientierten Methoden und beschränkt sich auf die Definition von Schnittstellen für die Module.

Mobile kann durch Verwendung einer interpretierenden Workflow-Maschine flexibel Workflow-Schemaänderungen umsetzen. Die Anforderung der dynamischen Erweiterbarkeit erreicht es jedoch nicht. Die Erweiterbarkeit ist nur während des Entwurfs gegeben, da für Erweiterungen eines Mobile-WfMS der gesamte Entwicklungszyklus, d.h. Quellcodeveränderung, Übersetzung, Installation usw., erneut durchlaufen werden muß. Erschwerend kommt hinzu, daß Erweiterungen nur bei Kenntnis des Quellcodes möglich sind. Die Zuweisung von informationstechnischen Ressourcen ist nur statisch auf der Ebene der sehr grobgranularen Module möglich. D.h. möchte man die informationstechnische Umsetzung eines Aspektes ändern, so muß das Modul neu erstellt werden. Die erforderliche feingranulare und dynamische Zuweisung von informationstechnischen Ressourcen ist mit Mobile nicht möglich. Durch die gewählte Architektur eines zentralen Koordinationsmoduls für die Koordination und Kommunikation zwischen den Aspektmodulen kommt es zu einem Leistungsengpaß, so daß die Skalierbarkeit von Mobile sehr gering ist. Diese Problematik wird auch in [Schu98] nicht erkannt, weswegen auch für die dort durchgeführte Verfeinerung die hier identifizierten Einschränkungen gelten.

4.2.6 WIDE

Das WIDE-Projekt [WIDE], [CeGS97], [CGPP97] hat das Ziel, auf der Basis von verteilten, aktiven Mechanismen eine verbesserte Unterstützung für Workflows zu schaffen. Basierend auf CORBA sollen Mechanismen für die Unterstützung von Regeln geschaffen werden, die wiederum zur Workflow-Unterstützung herangezogen werden sollen.

WIDE ermöglicht durch Regeländerungen die flexible Umsetzung von Schemaänderungen und ist nur statisch erweiterbar. Allein für den Operationsaspekt ist die Zuweisung informationstechnischer Ressourcen feingranular und zur Laufzeit möglich. CORBA sorgt für eine transparente Integration. Zur Erhöhung der Skalierbarkeit verwendet WIDE einen partitionierenden Ansatz und es wird eine Hierarchie von Workflow-Maschinen gebildet. Die Verwendung einer einzelnen Workflow-Maschine an der Spitze der Hierarchie schafft jedoch einen Leistungs- und Zuverlässigkeitsengpaß. Ein weiterer Engpaß ist die zentralisierte Regelverarbeitung sowie die zentralisierte Speicherung von Prozeßzustandsinformationen in einer Datenbank.

4.3 Ansätze zur Unterstützung weitreichender Prozesse

4.3.1 WorCos

Eine Reihe von Ansätzen versucht die Unterstützung von weitreichenden Prozessen auf der Basis von CORBA zu erreichen, wie beispielsweise [DKMS96], [MiSc98] und [VoWe99]. Der am weitesten entwickelte Ansatz ist der WorCos-Ansatz [Schu97a], [Schu97c], [Schu99]. Er wird herausgegriffen, da die für ihn angestellten Überlegungen auch auf die anderen Ansätze anwendbar sind. Zentrales Ziel des WorCos Ansatzes ist die Schaffung eines zur Object Management Architecture (OMA) [OMG] kompatiblen Workflow-Dienstes, der den Anforderungen der Object Management Group (OMG) [OMG] genügt. In [SBMW96] und [Böhm97] wird ein Vorschlag für eine CORBA-Facility [OMG] zur Unterstützung von Workflows unterbreitet.

Im WorCos Projekt soll eine tiefe Integration mit CORBA durch intensive Nutzung der CORBA-Services [OMG] erreicht werden. Viele andere Ansätze sehen in CORBA lediglich eine generische Middleware, um die technische Heterogenität zu überwinden. Grundlage der Workflow-Unterstützung von WorCos ist eine durchgängig objektorientierte Sichtweise. So werden Workflows in WorCos als CORBA-Objekte dargestellt, die durch Übersetzung gewonnen werden. Sie verfügen über Schnittstellen für folgende Dienste:

- Akteurzuordnung
- Zustandsmanipulation
- Zustands- und Ergebnisabfrage
- Abfrage von Metainformationen
- Technische Informationen

Es wird deutlich, daß sich eine große Menge von Funktionalität in den Workflow-Objekten befindet. Diese Gestaltung der Workflow-Objekte hat aber erhebliche Einschränkungen der Flexibilität und Erweiterbarkeit von WorCos zur Folge. Workflow-Schemaänderungen können nur durch Neuerstellung und Austausch der Workflow-Objekte umgesetzt werden. Auch liegt nur eine statische Erweiterbarkeit vor, da Erweiterungen nur durch Neuerstellung von Workflow-Objekten eingebracht werden können. Die Autarkie bezüglich der informationstechnischen Ressourcen zur Umsetzung der Workflows beschränkt sich auf den Operations- und Organisationsaspekt. So greifen die Workflow-Objekte bei ihrer Ausführung auf sogenannte „Human Resource Business Objects“ und „Resource Business Objects“ zu. „Human Resource Business Objects“ stellen den Kontakt zu menschlichen Benutzern her, während „Resource Business Objects“ Geschäftsobjekte kapseln. Diese können durch die Verwendung von CORBA transparent genutzt werden. Für andere Aspekte, wie beispielsweise den Kontrollaspekt, gibt WorCos die informationstechnischen Ressourcen statisch und grobgranular vor und verletzt so die Forderung nach Autarkie. Herauszuheben ist allerdings die hohe Skalierbarkeit von WorCos, die es durch eine schemaverteilte Ausführung, gemäß der Unterscheidung in Abschnitt 2.3.2.2, von Workflows erreicht.

4.3.2 CrossFlow

Das CrossFlow Projekt [CROS], [LuHo99] hat die Unterstützung von firmenübergreifenden Workflows zum Ziel. Dies soll erreicht werden, indem nicht mehr nur die Interoperabilität zwischen WfMS unterstützt wird, sondern auch weitergehende Fragestellungen verfolgt werden, die sich aus der Autarkie von Prozeßteilnehmern ergeben. Zur Lösung des Problems wird ein Konzept zur Integration externer Dienste über Gateways vorgeschlagen. Es wird also die Verbesserung der Nutzung dieser Dienste durch konventionell aufgebaute WfMS verfolgt, nicht je-

doch die Architektur von WfMS an sich neu konzipiert. In CrossFlow wird nicht von der grundsätzlichen Struktur von WfMS abgewichen, wie sie beispielsweise durch den WfMC-Ansatz konzipiert ist. Daher bleiben auch die grundsätzlichen Probleme der verwendeten WfMS wie eine eingeschränkte Skalierbarkeit bestehen.

Ein Gateway-Ansatz, wie der von CrossFlow, kann zwar die Integration von externen Diensten erleichtern. Er erweitert aber nicht das Spektrum der Dienste, für die dies möglich ist. Ebenso stellt CrossFlow keine Überlegungen bezüglich der Gestaltung von WfMS an, um deren Erweiterbarkeit zu gewährleisten. Flexibilität bezüglich der Umsetzung von Änderungen an Workflow-Schemata besteht im CrossFlow-Ansatz nur soweit dies schon innerhalb des verwendeten WfMS der Fall ist.

4.3.3 Mentor

Die im Mentor-Projekt [WKMS95], [WDMS95], [Wodt96], [WWWK96], [WoWe97], [WeMW98] entwickelte Architektur zielt auf unternehmensweite Unterstützung von Workflows. Ausgangspunkt ist die Modellierung der Workflows durch State- und Activity-Charts [Hare88], die eine leichte Partitionierung ermöglichen. Ausprägungen verschiedener Schemata können auf verschiedenen Systemen ausgeführt werden. Die Ausführung der Ausprägungen geschieht jedoch zentral durch Interpretation der State- und Activity-Charts. Es tauchen also erneut zentralisierte Elemente auf, die einen Leistungs- und Zuverlässigkeitsengpaß darstellen und die Skalierbarkeit des Ansatzes begrenzen.

Jeder Mentor-Server besteht nicht nur aus dem Interpreter für State-und-Activity-Charts, der als Workflow-Maschine fungiert, sondern auch aus einem CORBA-basierten Object Request Broker und einem TP-Monitor. Aufgabe des Object Request Brokers ist im wesentlichen die Integration von externen Diensten zur Unterstützung von Workflow-Operationen. Auf diese Weise kann eine teilweise Verteilung der Ausprägungsausführung erreicht werden. Mit Hilfe eines TP-Monitors werden Änderungen an Workflow-Schemata und –Daten über mehrere Server hinweg koordiniert.

Die hohen Anforderungen bestehend aus der Verwendung der Workflow-Maschine, eines TP-Monitors und eines Object Request Brokers für jeden Mentor-Server führten zur Entwicklung von Mentor-lite [MWGW98], [WWWK96]. Mentor-lite versucht, ein Workflow-System mit geringeren Anforderungen bereitzustellen. Die Workflow-Maschine besteht nur aus einem Interpreter für State-und-Activity-Charts, sowie einem ORB zum Zugriff auf externe Anwendungen. Erweiterungen werden ihrerseits wieder als Workflow implementiert. So sollen die Verwaltung von Arbeitslisten, die Verwaltung der Workflow-Historie und das Workflow Monitoring als Workflow dargestellt werden.

Mentor und Mentor-lite erreichen eine hohe Flexibilität. In bezug auf die Erweiterbarkeit ergibt sich jedoch ein zweigeteiltes Bild. Während Mentor kaum über Möglichkeiten in dieser Richtung verfügt, besitzt der Mentor-lite Ansatz die Möglichkeit, Erweiterungen als Workflows darzustellen und durch die Workflow-Maschine zu interpretieren. Auf diese Weise können in Mentor-lite zusätzliche Aspekte und Aspektelemente wie das Workflow-Monitoring oder die Workflow-Historie hinzugefügt werden. Allerdings wird der Nutzen dieses Verfahrens durch die dabei stattfindende Schachtelung von Interpretationsvorgängen gemindert. In Mentor-lite werden Erweiterungen der Workflow-Schemata selbst wiederum interpretativ ausgeführt. Mentor-lite ist daher ein gutes Beispiel für die Unabhängigkeit der Eigenschaften der Erweiterbarkeit und der flexiblen Zuweisung informationstechnischer Ressourcen. Obwohl mit dem in Mentor-lite vorgesehenen Mechanismus Erweiterbarkeit gegeben ist, ist die dafür verwendete Ressource in Form der Workflow-Maschine festgelegt. Es zeigt sich, daß die Erweiterbarkeit nicht die Autarkie bei der Zuweisung von informationstechnischen Ressourcen umfaßt. So erfolgt die

Umsetzung der Erweiterungen stets mit der State-und-Activity-Charts verarbeitenden Workflow-Maschine, die Verwendung anderer Ressourcen für die Umsetzung von Erweiterungen ist nicht möglich. Mentor und Mentor/lite unterstützen nur eine Partitionierung, eine Verteilung von Diensten ist nur für den Operationsaspekt möglich.

4.3.4 Meteor

Das Meteor- und Meteor$_2$-Projekt [Wang95], [MSKW96] hat zum Ziel, eine voll verteilte Workflow-Unterstützung auf der Basis von CORBA für den unternehmensweiten Einsatz bereitzustellen. In [DKMS96], [MPSK98] wird eine Anbindung an das WWW vorgestellt.

Aufgabenverwalter (Task Manager), dargestellt in Abbildung 32, sind das zentrale Element von Meteor. Die Aufgabenverwalter steuern die eigentlichen Aufgaben (Tasks) und die dazugehörige Benutzungsschnittstelle. Aufgabenverwalter bestehen aus der Aufgabenaktivierung (task activation), Aufgabenaufruf (task invocation), Überwachung (observer), Fehlerbehandlung und Wiederherstellung. Die Aufgabenverwalter sind für die Steuerung des Kontrollflusses zuständig. Sie werden durch Übersetzung der Kontrollflußanteile des Workflow-Schemas gewonnen. Es wird aber nicht für deren flexible Ersetzbarkeit gesorgt, so daß es erneut zu hohen Aufwänden bei der Umsetzung von Workflow-Schemaänderungen kommt.

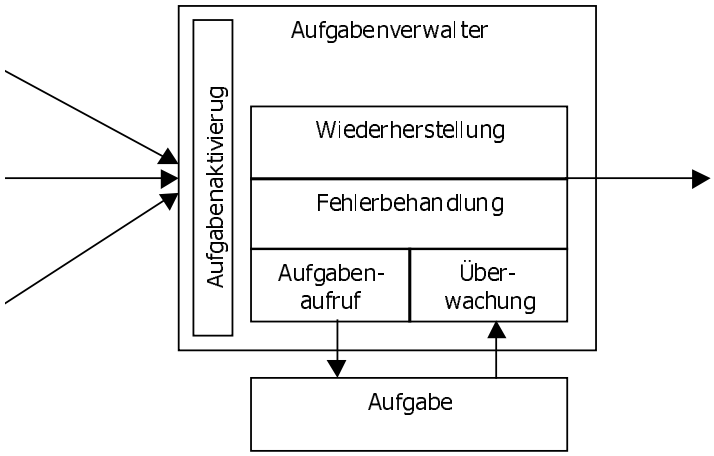


Abbildung 32: Aufgabenverwalter in Meteor

Die Aktivierung eines Aufgabenverwalters geschieht durch Aufruf der Aufgabenaktivierung mit den Parametern, die zur Ausführung der Aufgabe notwendig sind. Hierzu gehören insbesondere Daten und Verweise auf Daten, die bearbeitet werden sollen. Der Prozeß der Aufgabenaktivierung zerfällt in drei Teile: Teile vor, während, und nach der Aufgabenausführung. Im ersten Teil werden Bedingungen definiert, die die Aufgabenaktivierung vom Abschluß von Vorgängeraufgaben abhängig machen. Diese können durch Und- bzw. Oder-Operatoren verknüpft sein. Nach Erfüllung einer solchen Bedingung werden die als Parameter angefügten Daten entpackt und die Aufgabe gestartet. Nach deren Ende werden die Daten für die Weiterverarbeitung in den folgenden Aufgabenverwaltern vorbereitet. Auf die Details der Wiederherstellung, Fehlerbehandlung usw. soll an dieser Stelle nicht weiter eingegangen werden.

Die Umsetzung von Workflows in Meteor geschieht durch Umsetzung der Workflow Intermediate Language (WIL) in miteinander verknüpfte Aufgabenverwalter und Aufgaben. Dargestellt ist dies in Abbildung 33. Dabei werden zwei wichtige Unterschiede von Meteor zu den meisten anderen Ansätzen deutlich. Die Task Manager sind unabhängige Einheiten, die auf unterschiedlichen Systemen plaziert sein können. Außerdem werden sie generiert, d.h. aus der WIL wird

direkt ausführbarer Code erzeugt. Es findet also keine Interpretation wie in den meisten WfMS statt.

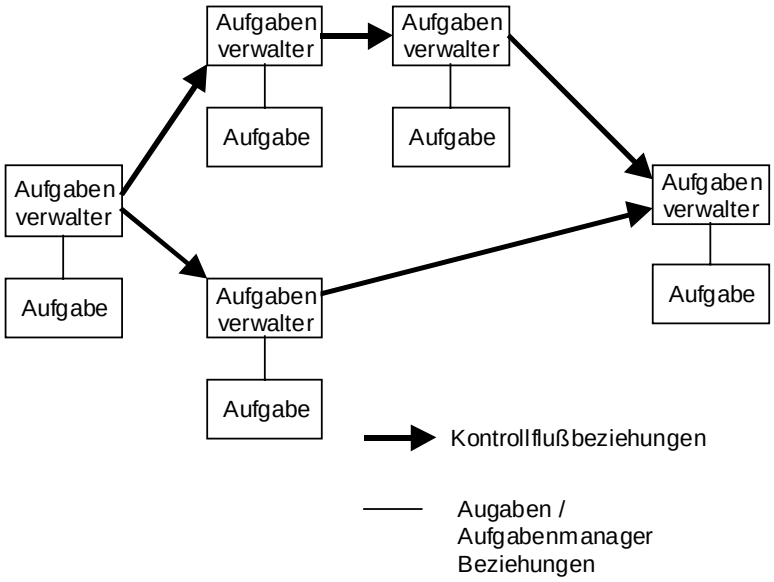


Abbildung 33: Workflow-Umsetzung in Meteor

Das angewandte Konzept der Codegenerierung für die Umsetzung der Workflows schränkt die Flexibilität des Meteoransatzes stark ein, da sie Änderungen der Workflow-Schemata sehr aufwendig macht. Zusätzlich erhöht wird der Aufwand durch den Umstand, daß der Aufgabenverwalter nicht nur für den Kontrollfluß, sondern auch den Datenfluß, die Fehlerbehandlung und Wiederherstellungsmaßnahmen zuständig ist. Daher hat jede Änderung an einem dieser Aspekte eine Neuerstellung und –verteilung des gesamten Aufgabenverwalters zur Folge, obwohl eigentlich nur ein Aspekt von ihm betroffen ist. Erweiterungen um zusätzliche Aspekte oder Aspektelemente erfordern Erweiterungen der Generierungsmechanismen für Aufgabenverwalter, was aber kaum durch den Benutzer von Meteor durchgeführt werden könnte. Daher kann kaum von einer Erweiterbarkeit gesprochen werden. Meteor kann alleine Elemente des Operationsaspekts durch die Aufgabenverwalter integrieren. Informationstechnische Ressourcen können nur im Rahmen des Operationsaspekts zugewiesen werden. Die Umsetzung der übrigen Aspekte ist in Meteor fest vorgegeben. Ein herausstechendes Merkmal von Meteor ist seine hohe Skalierbarkeit, die allerdings mit der oben beschriebenen Inflexibilität durch die Verwendung übersetzter Aufgabenverwalter erkauft wird.

4.3.5 Wide Area Group Flow

Die Ziele des Wide Area Group Flow Ansatzes [Riem98] sind stark betriebswirtschaftlich geprägt. So konzentriert er sich auf die Probleme, die sich durch die organisatorisch-rechtliche Autarkie von Teilnehmern weitreichender Prozessen ergeben. Im wesentlichen werden die organisatorischen Unterschiede bei der Unterstützung von Prozessen zwischen Unternehmen behandelt. Hierzu gehört beispielsweise die fehlende Möglichkeit einer zentralisierten Modellierung, Verwaltung und Unterstützung, da die Teilnehmer des Prozesses das Wissen um die Gestaltung ihrer Prozesse nicht nach außen geben möchten. Der Wide Area Group Flow-Ansatz ist stark auf die Verwendung von Lotus Notes, einer Groupware-Plattform, ausgerichtet. Seine Entwicklung ist daher in erster Linie vom Bestreben geprägt, die von Lotus Notes angebotenen Techniken für die Unterstützung der organisatorischen Besonderheiten bei unternehmensübergreifenden Workflows zu nutzen. So soll unter der Randbedingung eines möglichst geringen Informationsaustausches zwischen zwei Lotus Notes-Systemen dennoch die Unter-

stützung eines übergreifenden Workflow gewährleistet werden. Zu diesem Zweck sind im Rahmen der Möglichkeiten von Lotus Notes spezielle Notes-Dokumente entwickelt, sowie Notes-Datenbanken entworfen worden. Durch ihre Replikation sollen die zur Prozeßausführung notwendigen Informationen in beiden Unternehmen zur Verfügung stellen. Eine Nutzung von Umgebungsdiensten ist daher nicht möglich.

Der Wide Area Group Flow-Ansatz erlaubt die flexible Änderung von Workflow-Schemata allerdings nur innerhalb der von seinem Workflow-Modell vorgegebenen Grenzen. Erweiterungen sind im Wide Area Group Flow-Ansatz weder für Aspekte noch Aspektelemente möglich. Der Wide Area Group Flow-Ansatz läßt, bis auf Anwendungsprogramme für die Unterstützung von Workflow-Operationen, keine Autarkie bei der informationstechnischen Umsetzung zu. Wie etliche andere Ansätze führt der Wide Area Group Flow-Ansatz keine wirkliche Verteilung, sondern nur eine Partitionierung von Workflows durch, was seine Skalierbarkeit begrenzt.

4.4 Zusammenfassung

Aufgabe dieses Kapitels war es, die These zu verifizieren, daß keiner der existierenden Ansätze zur Unterstützung weitreichender Prozesse in der Lage ist. Als Kriterium für die Beurteilung der existierenden Ansätze dienen die in Kapitel 3 identifizierten Anforderungen aus der Unterstützung weitreichender Prozesse. Für die Durchführung der Untersuchung wurden die Ansätze zunächst danach unterteilt, ob sie Standardisierungsansätze sind, einzelne Anforderungen aus der Unterstützung weitreichender Prozesse adressieren oder das Problem der Unterstützung weitreichender Prozesse in seiner Gesamtheit zu lösen versuchen. Bei der Untersuchung der Ansätze stellte sich heraus, daß sie zwar durchaus einzelne Anforderungen erfüllen. Kein Ansatz ist jedoch in der Lage, die Anforderungen aus der umfassenden Unterstützung weitreichender Workflows in ihrer Gesamtheit zu erfüllen. Aufgabe des folgenden Kapitels wird es daher sein, die Ursachen hierfür zu ermitteln.

5 Lösungsansatz

Die Untersuchung existierender Ansätze hat gezeigt, daß diese jeweils nur Teilmengen der Anforderungen aus der Unterstützung weitreichender Workflows erfüllen. In diesem Kapitel wird eine Analyse der Ursachen hierfür durchgeführt und ein Lösungsansatz entwickelt. Aufgabe der Analyse ist es, die Problembereiche existierender Ansätze bei der Unterstützung weitreichender Prozesse zu identifizieren.

Die Herausforderung bei der Entwicklung des Lösungsansatzes ist es, eine geeignete Lösungsstrategie zu entwickeln. Diese beinhaltet mehrere Schritte, die für die identifizierten Problembereiche Lösungen schaffen. Zwischen den Lösungsschritten bestehen Abhängigkeiten in Form von funktionalen Anforderungen. Es gilt daher die Lösungsstrategie so zu gestalten, daß diesen Anforderungen genüge getan wird. Zusätzlich leiten sich aus den in den drei Lösungsschritten entwickelten Konzepten nicht-funktionale Anforderungen ab, die von einer Software-Architektur erfüllt werden müssen, die diese Konzepte umsetzen soll. Erst wenn eine Software-Architektur als Realisierungsgrundlage geschaffen worden ist, die diese Anforderungen erfüllt, kann als letzter Schritt die Realisierung der Lösungskonzepte durchgeführt werden.

5.1 Analyse

In diesem Abschnitt wird für jede Anforderung untersucht, worin die Ursachen dafür liegen, daß sie von den existierenden Ansätzen nicht, oder nur unter Nichterfüllung einer anderen Anforderung erfüllt werden kann. Tabelle 2 faßt die Erfüllung der Anforderungen durch die existierenden Ansätze zusammen. Für die Anforderung der Autarkie werden die Aspekte angegeben, für die diese erfüllt wird. Das Kürzel op steht für den Operationsaspekt, das Kürzel og für den Organisationsaspekt. Die letzte Spalte gibt das Verteilungsmodell des Ansatzes wieder. In dieser Spalte wird nach zentralisierten, ausprägungsverteilten und schemaverteilten Ansätzen unterschieden. Drei Ansätze sind grau hinterlegt, die dadurch auffallen, daß sie im Gegensatz zu den übrigen die Anforderung der Skalierbarkeit erfüllen, nicht aber die der Flexibilität.

	Autarkie	Flexibilität	Erweiter-barkeit	Skalier-barkeit	Verteilungsmodell
Broker/Services / Eve	op, og	ja	nein	nein	schemaverteilt
CrossFlow	op	ja	nein	nein	zentralisiert
Exotica / FMQM	op	nein	nein	ja	ausprägungsverteilt
INCOME/WF	op	ja	nein	nein	zentralisiert
Mentor	op	ja	nein	nein	zentralisiert
Mentor-lite	op	ja	ja	nein	zentralisiert
Meteor	op	nein	nein	ja	schemaverteilt
MetuFlow	op	nein	nein	nein	zentralisiert
Mobile	op	ja	nein	nein	zentralisiert
WfMC	op	ja	nein	nein	zentralisiert
WIDE	op	ja	nein	nein	zentralisiert
Wide Area Group Flow	op	ja	nein	nein	zentralisiert
WorCos	op, og	nein	nein	ja	schemaverteilt

Tabelle 2: Vergleich der existierenden Ansätze

5.1.1 Autarkie bei der informationstechnischen Umsetzung

Die Autarkie ist bei den existierenden Ansätzen meist auf die Zuweisung von Ressourcen zur Unterstützung des Funktionsaspekts im Rahmen des Operationsaspekts begrenzt. Die Ursache hierfür findet sich, wenn man sich das Workflow-Metamodell dieser Ansätze betrachtet. Mit Ausnahme des Funktions- und Operationsaspekts wird nicht zwischen einer spezifizierenden, logischen Ebene und einer implementierenden, physischen Ebene unterschieden. Daher ist eine feingranulare und dynamische Zuweisung informationstechnischer Ressourcen schon auf der Ebene des Workflow-Metamodells versperrt.

Dies wird am Beispiel der Referenzarchitektur der Workflow-Management Coalition veranschaulicht. In Abbildung 34 ist die Zuordnung von Ressourcen zu Aspekten dargestellt. Man sieht, daß alle Aspekte, außer dem Operationsaspekt, durch die Workflow-Maschine umgesetzt werden. So können zwar im Informationsaspekt Datenstrukturen festgelegt werden. Die zur Speicherung der Daten verwendete Datenbank kann jedoch nicht bestimmt werden, sondern wird vom Entwickler des WfMS a priori festgelegt. Auch für die übrigen Aspekte ist das WfMS fest als Ressource zur Unterstützung vorgesehen. Hierdurch wird deutlich, daß den existierenden Ansätzen die autarke Zuweisung informationstechnischer Ressourcen durch das ihnen zugrundeliegende Workflow-Metamodell prinzipiell versperrt ist.

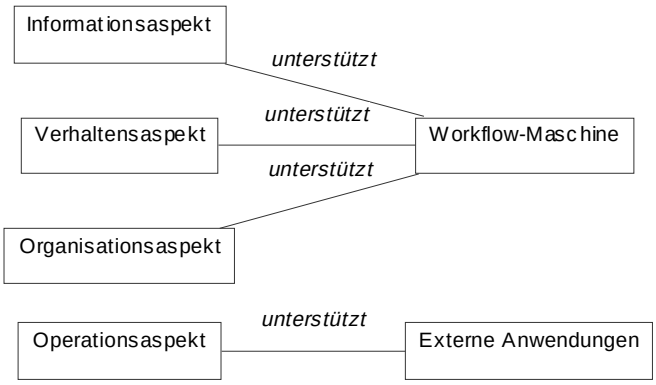


Abbildung 34: Ressourcenzuweisung im WfMC-Referenzmodell

5.1.2 Flexibilität und Skalierbarkeit

Bei der Untersuchung der Flexibilität und Skalierbarkeit läßt sich beobachten, daß die Ansätze in zwei Gruppen zerfallen. Die erste Gruppe, zu der der Großteil der Ansätze gehört, besitzt zwar eine hohe Flexibilität gegenüber Schemaänderungen. Ihre Skalierbarkeit ist jedoch sehr eingeschränkt, da sie nur eine Partitionierung erreichen, nicht jedoch eine Schema- oder gar Ausprägungsverteilung. Die zweite Gruppe ist deutlich kleiner und umfaßt den Meteor, Exotica und WorCos Ansatz (sie sind in Tabelle 2 grau unterlegt). Die Ansätze erreichen zwar eine gute Skalierbarkeit, aber ihre Flexibilität ist sehr gering. Dieser Zusammenhang läßt sich erklären, wenn man die Bildung von Ausprägungen untersucht. Hierfür gibt es bisher zwei Möglichkeiten, die in Abbildung 35 dargestellt sind.

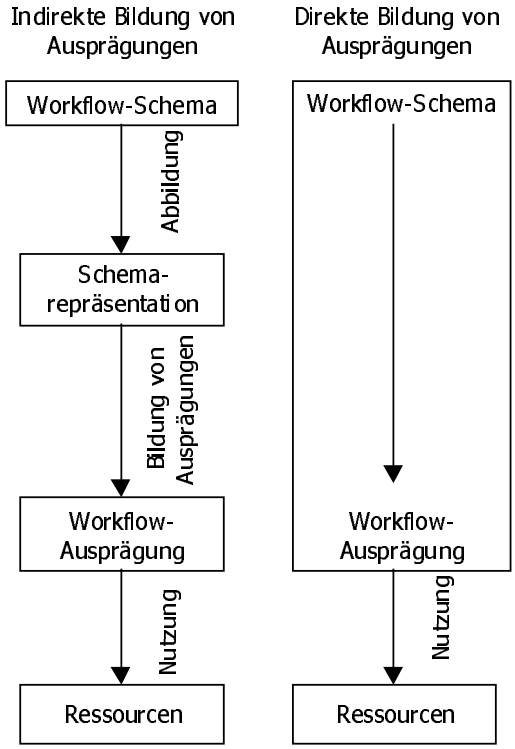


Abbildung 35: Indirekte und direkte Ausprägungsbildung von Workflow-Schemata

Der Meteor, Exotica und der WorCos Ansatz führen die indirekte Bildung von Ausprägungen durch. Bei ihr wird die Bildung von Workflow-Ausprägungen in zwei Schritten durchgeführt.

Zunächst wird ausgehend vom Workflow-Schema eine Schemarepräsentation geschaffen. Bei WorCos erfolgt dies, indem das Workflow-Schema in eine Workflow-Klasse umgesetzt wird. In einem zweiten Schritt werden auf der Basis dieser Schemarepräsentation Workflow-Ausprägungen gebildet. Dies geschieht in WorCos, indem Ausprägungen der im ersten Schritt gebildeten Workflow-Klasse geschaffen werden. Wichtig ist dabei die zweifache Bildung von Ausprägungen zu erkennen. So werden Ausprägungen des Workflow-Schemas gebildet, genauso wie Ausprägungen der Schemarepräsentation gebildet werden, allerdings auf software-technischer Ebene.

Bei der indirekten Bildung von Ausprägungen können die Erzeugung der Schemarepräsentation und die Bildung von Ausprägungen nicht nur zeitlich, sondern auch örtlich getrennt ablaufen. Die Ausprägungsbildung kann daher zu einem anderen Zeitpunkt und auf einem anderen System ausgeführt werden. Durch Verteilung der Schemarepräsentationen auf unterschiedliche Rechner kann eine Schemaverteilung erreicht werden. Außerdem ist es möglich, die Schemarepräsentation zu vervielfältigen und so auf mehreren Rechnern parallel Workflow-Ausprägungen zu bilden. Auf diese Weise kann die Ausprägungsverteilung erreicht werden.

Die Möglichkeit, die Bildung der Schemarepräsentation von der Bildung von Ausprägungen örtlich und zeitlich zu trennen, und die sich daraus ergebende Möglichkeit zur Schema- oder gar Ausprägungsverteilung begründen die gute Skalierbarkeit von Ansätzen, die auf der indirekten Ausprägungsbildung beruhen. Die Verwendung einer Schemarepräsentation stellt allerdings einen Nachteil dar, wenn Schemaänderungen eingebracht werden, da die Schemarepräsentation nicht zur Laufzeit angepaßt werden kann. Hierdurch wird aber die Anforderung der Flexibilität verletzt. Die Ursache hierfür ist die monolithische und statische Gestaltung der Schemarepräsentation der existierenden Ansätze. Sie kann nicht zur Laufzeit flexibel an Schemaänderungen angepaßt werden, sondern muß komplett neu erstellt werden.

Im Gegensatz zu den indirekt Ausprägungen bildenden Ansätzen wird bei den direkt Ausprägungen bildenden Ansätze keine Schemarepräsentation erstellt. Zu ihnen gehören alle Ansätze außer der drei oben erwähnten. Direkt Ausprägungen bildende Ansätze erzeugen auf der Basis des Workflow-Schemas unmittelbar eine Workflow-Ausprägung. Änderungen des Workflow-Schemas können daher sofort bei der Bildung neuer Ausprägungen umgesetzt werden. Direkt Ausprägungen bildende WfMS interpretieren typischerweise die in einer Workflow-Sprache repräsentierten Workflow-Schemata.

Bei direkt Ausprägungen bildenden Ansätzen kann die Bildung der Ausprägungen nicht auf zwei Schritte aufgeteilt werden, wie bei den indirekt Ausprägungen bildenden Ansätzen. Daher entfällt auch die Möglichkeit die Bildung von Ausprägungen örtlich zu verteilen und es ist keine Schema- bzw. Ausprägungsverteilung erreichbar. Hierin begründet sich die schlechte Skalierbarkeit des direkt Ausprägungen bildenden Ansatzes. Aus diesen Überlegungen wird das Dilemma deutlich, indem sich die existierenden Ansätze befinden. Direkt Ausprägungen bildende WfMS besitzen zwar eine hohe Flexibilität, bezahlen jedoch hierfür mit einer stark eingeschränkten Skalierbarkeit. Auf der anderen Seite erkaufen die indirekt Ausprägungen bildenden WfMS ihre Skalierbarkeit mit einer stark eingeschränkten Flexibilität.

5.1.3 Erweiterbarkeit des Workflow-Modells

Die existierenden WfMS lassen bis auf eine Ausnahme keine Erweiterungen zur Laufzeit zu. Erweiterungen können nur durch Neuerstellung des WfMS eingebracht werden, was in den allerwenigsten Fällen praktikabel ist. Die einzige Ausnahme ist der Mentor-lite Ansatz, der Erweiterungen als Workflow implementiert und wie ein Workflow interpretativ ausführt. Dadurch gehört er aber zu den direkt Ausprägungen bildenden Ansätzen, die wie die Diskussion in Abschnitt 5.1.2 gezeigt hat, von ihrer Skalierbarkeit her begrenzt sind. Mentor-lite erreicht zwar

die Erweiterbarkeit, opfert dafür aber die Skalierbarkeit. Außerdem kann er nur die eingebaute Workflow-Maschine zur Umsetzung von Erweiterungen verwenden, die Nutzung anderer Ressourcen ist nicht möglich. Die tiefere Ursache für die Einschränkungen der existierenden Ansätze ist die statische und implizite Repräsentation des Workflow-Modell. Mit Ausnahme des Mentor-lite-Ansatzes ist das Workflow-Modell fest eingeprägt. Bei keinem Ansatz erfolgt eine explizite Darstellung des Workflow-Modells, es ist vielmehr implizit in der Struktur der Workflow-Maschine enthalten.

5.1.4 Fazit

Basierend auf der durchgeführten Analyse konnten drei Problembereiche identifiziert werden.

1. Das Workflow-Metamodell, das den existierenden Ansätzen zu Grunde liegt, verhindert die autarke Zuweisung von informationstechnischen Ressourcen.

2. Die Anforderung der dynamischen Erweiterbarkeit kann von den bisherigen Ansätzen nicht erfüllt werden, da ihnen ein a priori bestimmtes Workflow-Modell fest eingeprägt ist.

3. Die existierenden Ansätze befinden sich in einem Dilemma zwischen Flexibilität und Skalierbarkeit. Direkt Ausprägungen bildende Ansätze erreichen eine hohe Flexibilität bei eingeschränkter Skalierbarkeit. Indirekt Ausprägungen bildende Ansätze erreichen hingegen eine hohe Skalierbarkeit bei eingeschränkter Flexibilität. Ursache für die mangelnde Skalierbarkeit direkt Ausprägungen bildender Ansätze ist die zentralisierte Abbildung des Workflow-Schemas und die Zusammenfassung mit der Bildung von Ausprägungen. Die mangelnde Flexibilität indirekt Ausprägungen bildender Ansätze ist durch die inflexible Gestaltung der Schemarepräsentation begründet.

5.2 Lösungsansatz

In diesem Abschnitt wird zunächst eine Lösungsstrategie entwickelt, um dann detailliert die einzelnen Lösungsschritte darzustellen.

5.2.1 Lösungsstrategie

Ziel bei der Entwicklung der Lösungsstrategie muß es sein, die einzelnen Lösungsschritte in eine geeignete Reihenfolge zu bringen. Daher gilt es, Abhängigkeiten zwischen den einzelnen Lösungsschritten zu identifizieren und dementsprechend die Abfolge der Lösungsschritte zu gestalten. Bei diesen Abhängigkeiten kann es sich nicht nur um direkt ersichtliche funktionale Anforderungen handeln, sondern auch um nicht-funktionale Anforderungen. Diese erschließen sich nicht direkt, sondern werden nur durch eine tiefergehendere Analyse deutlich.

Zunächst können drei Lösungsschritte identifiziert werden, für die eine Reihenfolge bei der Lösungsentwicklung notwendig ist. Diese sind in Abbildung 36 dargestellt. Für die Realisierung dieser Lösungsschritte sind keineswegs beliebige Software-Architekturen geeignet. Die Kriterien, die über die Eignung entscheiden, sind nicht-funktionale Anforderungen, die sich aus diesen drei Lösungsschritten ergeben. Daher werden bei jedem der Lösungsschritte auch die sich aus ihm ableitenden nicht-funktionalen Anforderungen angegeben.

1. Zunächst ist ein Workflow-Metamodell zu entwickeln, das die Grundlage für Workflow-Modelle bildet, die eine autarke Zuweisung von informationstechnischen Ressourcen ermöglichen.

2. Im darauf folgenden Schritt ist eine Repräsentation von Workflow-Modellen zu entwickeln, die zur Laufzeit eine Erweiterung zuläßt. Als Vorgabe für die zu repräsentierenden Workflow-Modelle dient das im vorangegangenen Schritt konzipierte Workflow-Metamodell. Das Workflow-Metamodell gibt also funktionale Anforderungen vor.

3. Um dem identifizierten Dilemma zwischen Flexibilität und Skalierbarkeit zu entrinnen, wird ein indirekt Ausprägungen bildender Ansatz verfolgt. Dieser wird allerdings mit einer dynamisch anpaß- und erweiterbaren Schemarepräsentation versehen. Auf diese Weise wird eine flexible Umsetzung von Schemaänderungen erreicht. Durch die Verwendung eines indirekt Ausprägungen bildenden Ansatzes ist zudem die Skalierbarkeit des Ansatzes gegeben. Dieser Lösungsschritt stützt sich auf das im vorangegangenen Lösungsschritt geschaffene Workflow-Modell.

4. Aus den erhobenen nicht-funktionalen Anforderungen ergibt sich die Notwendigkeit für einen vierten Lösungsschritt, in dem zunächst die Realisierungsgrundlage für die entwickelten Lösungskonzepte entwickelt werden muß. Dazu muß eine Software-Architektur geschaffen werden, die die identifizierten nicht-funktionalen Anforderungen erfüllt.

5. Erst nach der Bereitstellung dieser Software-Architektur kann die Realisierung der Lösungskonzepte durchgeführt werden.

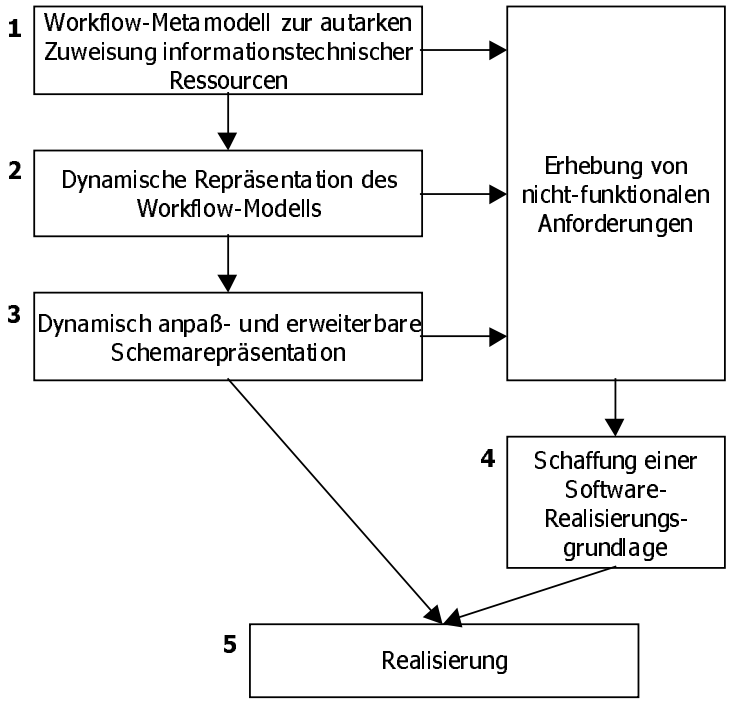


Abbildung 36: Lösungsstrategie

5.2.2 Lösungsschritte

Nun werden, der obigen Lösungsstrategie folgend, die einzelnen Lösungsschritte detaillierter dargestellt werden. Die ersten 3 Lösungsschritte sind das Zwei-Ebenen-Workflow-Metamodell, das Workflow-Modell-Wörterbuch und die Aspektelementorientierte Schemarepräsentation: Bei jedem dieser Lösungsschritte wird auch angegeben, welche nicht-funktionalen Anforderungen sich aus ihm ergeben.

Im vierten Lösungsschritt wird mit Komponentensystemen eine Software-Architektur vorgestellt, die die ermittelten nicht-funktionalen Anforderungen erfüllt. Der fünfte Lösungsschritt führt Komponentensysteme und die entwickelten Lösungskonzepte in Form der aspektorientierten Komponentensysteme zusammen.

5.2.2.1 Zwei-Ebenen-Workflow-Metamodell

Grundlage für das weitere Vorgehen ist die Schaffung eines Workflow-Metamodells, das Workflow-Modelle beschreibt, die die dynamische und feingranulare Zuweisung von informationstechnischen Ressourcen zur Prozeßunterstützung erlauben. Aus dem Zwei-Ebenen-Workflow-Metamodell ergibt sich die nicht-funktionale Anforderung der Ressourcenkonfigurierbarkeit. Das Zwei-Ebenen Workflow-Metamodell wird in Kapitel 6 vorgestellt.

5.2.2.2 Workflow-Modell-Wörterbuch

In einem zweiten Schritt, der in Kapitel 7 dargestellt ist, wird auf der Basis dieses Workflow-Metamodells ein Workflow-Modell-Wörterbuch konzipiert. Dieses ermöglicht es, das Workflow-Modell eines WfMS dynamisch erweiterbar machen: Zusätzliche Workflow-Modellelemente können zur Laufzeit aufgenommen werden. Aus dem Workflow-Modell Wörterbuch ergeben sich die nicht-funktionalen Anforderungen der Dienst- und Ressourcenerweiterbarkeit. Auf der Basis der im Workflow-Modell-Wörterbuch abgelegten Workflow-Modelle können Workflow-Schemata gebildet werden, die durch die im folgenden Lösungsschritt zu schaffende aspektelementorientierte Schemarepräsentation umgesetzt werden können.

5.2.2.3 Aspektelementorientierte Schemarepräsentation

Den dritten Schritt bildet die in Kapitel 8 wiedergegebene Schaffung einer aspektelementorientierten Schemarepräsentation, die zur Laufzeit anpaß- und erweiterbar ist und so Schemaänderungen flexibel umsetzen kann. Aus ihr ergeben sich die nicht-funktionalen Anforderungen der Dienste- und Anwendungskonfigurierbarkeit sowie der Anwendungserweiterbarkeit.

5.2.2.4 Komponentensysteme als Realisierungsgrundlage

Um diese Lösungskonzepte umsetzen zu können, wird mit Komponentensystemen eine Architektur geschaffen, die in der Lage ist, die Gesamtheit der identifizierten nicht-funktionalen Anforderungen zu erfüllen. Dies geschieht in Kapitel 9. Komponentensysteme setzen sich aus Komponenten, komponentenorientierten Rahmenwerken und kompositen Anwendungen zusammen. Jeder dieser drei Bestandteile erfüllt eine oder mehrere der nicht-funktionalen Anforderungen.

5.2.2.5 Aspektorientierte Komponentensysteme als Realisierung

Mit der Schaffung der Komponentensysteme wurde die Grundlage für die Realisierung der obigen Lösungskonzepte gelegt. Daher kann in Kapitel 10 beschrieben werden, wie die aspektelementorientierte Schemarepräsentation und das Workflow-Modell-Wörterbuch mit Hilfe von Komponentensystemen realisiert werden. Da als Granulat für die Bildung von Komponenten Aspekte verwendet werden, werden die entstehenden Systeme als aspektorientierte Komponentensysteme bezeichnet.

Das Zusammenspiel der Lösungsschritte ist in Abbildung 37 dargestellt. Dabei ist auch angegeben, welche der informationstechnischen Anforderungen aus der Unterstützung weitreichender Prozesse durch den jeweiligen Lösungsschritt erfüllt werden. Dies geschieht durch eine gestrichelte Umrandung der Lösungskonzepte, die eine der Anforderungen, Autarkie, Erweiterbarkeit sowie Flexibilität und Skalierbarkeit erfüllen.

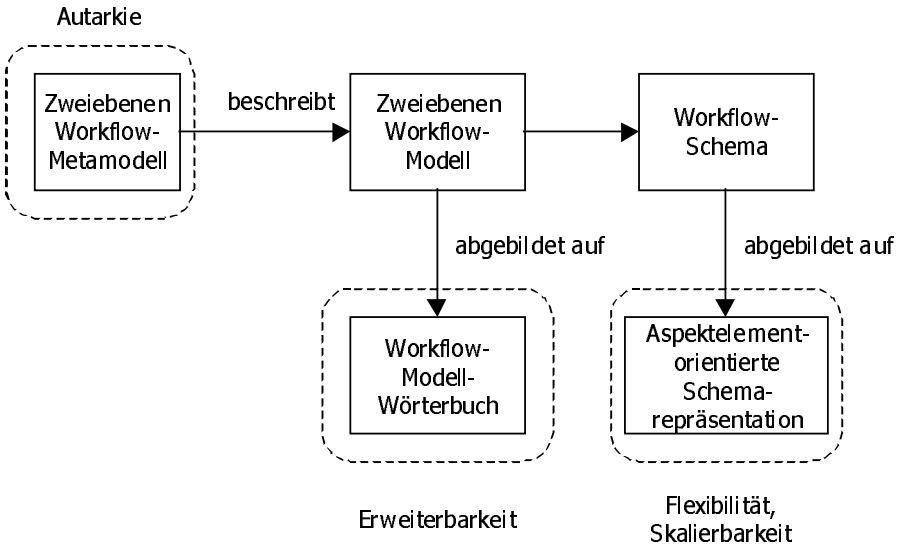


Abbildung 37: Lösungsschritte

5.3 Zusammenfassung

Dieses Kapitel hatte zwei Ziele. Erstens sollten die Ursachen dafür aufgedeckt werden, daß die existierenden Ansätze nur einzelne Anforderungen aus der Unterstützung weitreichender Prozesse erfüllen. Zweitens sollte basierend auf dieser Analyse ein Lösungsansatz bestimmt werden.

Die Analyse der existierenden Ansätze brachte drei Problembereiche zu Tage, deren Gestaltung die existierenden Ansätze an der Erfüllung der Anforderungen aus der Unterstützung weitreichender Prozesse hindert. Es sind dies die Gestaltung des Workflow-Metamodells, die feste Einprägung des Workflow-Modells und die inflexible Repräsentation von Workflow-Schemata.

Auch das zweite Ziel des Kapitels wurde erreicht. So wurde ein Lösungsansatz zur Bewältigung der obigen Problembereiche entwickelt, der die Grundlage für die folgenden Kapitel legt. Dieser Lösungsansatz besteht aus drei Lösungskonzepten, nämlich dem Zwei-Ebenen-Workflow-Metamodell, dem Workflow-Modell-Wörterbuch und der Aspektelementorientierten Schemarepräsentation. Zu deren Entwicklung wurde ein Lösungsstrategie entworfen, die auf der Basis funktionaler Anforderungen eine Reihung der Lösungskonzepte in Lösungsschritten durchführte. Dabei konnte die Notwendigkeit eines zusätzlichen Lösungsschrittes erkannt werden. Aus jedem Lösungskonzept ergeben sich nicht-funktionale Anforderungen, die von einer Software-Architektur erfüllt werden müssen, wenn sie zur Umsetzung der Lösungskonzepte verwendet werden soll. Der vierte Lösungsschritt besteht daher in der Bereitstellung einer solchen Software-Architektur in Form von Komponentensystemen. Auf ihrer Basis kann im fünften Lösungsschritt die Realisierung der Lösungskonzepte durchgeführt werden.

6 Zwei-Ebenen-Workflow-Metamodell

In diesem Kapitel wird ein Workflow-Metamodell geschaffen, das die dynamische und feingranulare Zuweisung informationstechnischer Ressourcen für die Prozeßunterstützung ermöglicht. Hierzu sollen zunächst die bisherigen Ansätze zur Gestaltung von Workflow-Metamodellen dahingehend untersucht werden, ob sie Lösungsbeiträge für das hier zu lösende Problem erbringen können.

Da dies nicht der Fall ist, wird das Zwei-Ebenen-Workflow-Metamodell konzipiert. Die Herausforderung bei seiner Gestaltung liegt darin, nicht nur die dynamische, sondern auch eine geeignet feingranulare Zuweisung von informationstechnischen Ressourcen sicherzustellen. Zur Veranschaulichung wird ein Beispiel für die Anwendung des Zwei-Ebenen-Workflow-Metamodells gegeben. Abschließend wird eine Bewertung der erzielten Ergebnisse durchgeführt, die klärt, ob das Ziel des Kapitels erreicht wurde.

6.1 Existierende Ansätze

Umfangreiche Überlegungen zur Gestaltung von Workflow-Metamodellen finden sich in [DeGS95], [Jabl95a], [Wesk96], [CCPP96b], [HoSW96], [Ober96a], [JaBS97] und in [Schu99]. Dabei stehen jedoch modellierungstheoretische Überlegungen im Vordergrund. Ziel ist es, mit Hilfe des Workflow-Metamodells eine klare Konzeption von Workflow-Modellen zu erreichen. Die Ergebnisse dieser Arbeiten sind in Kapitel 2 vorgestellt worden und flossen in mehrere Ansätze ein, die in Kapitel 4 untersucht wurden.

Andere Ansätze zur Gestaltung von Workflow-(Meta-)Modellen konzentrieren sich auf die Unterstützung von Ausnahmesituationen. So wird in [CGPP97] und in [CVSG96] untersucht, wie ein Workflow-Metamodell für die Ausnahmebehandlung optimiert werden kann. Auch in [ChLK99] wird dieser Frage nachgegangen. Dabei wird insbesondere die Umsetzung mit Hilfe von objektorientierten und aktiven Mechanismen untersucht.

Ansätze, die sich mit der Gestaltung von Workflow-Metamodellen in Hinblick auf die dynamische und feingranulare Zuweisung von Ressourcen beschäftigen, sind nicht vorhanden. Der einzige Bereich, in dem einige Ansätze eine gewisse Flexibilität bei der Ressourcenzuordnung erreichen, ist die Zuweisung von Anwendungsprogrammen, wie die Analyse in Kapitel 5 gezeigt hat. Sie ist jedoch auf diesen Aspekt begrenzt. Zusammenfassend ist daher festzustellen, daß kein existierender Ansatz zur Gestaltung von Workflow-Metamodellen die Anforderung der flexiblen und feingranularen Zuweisung informationstechnischer Ressourcen erfüllt.

6.2 Konzeption

6.2.1 Logische und physische Ebene

Um die flexible Zuweisung von Ressourcen zu ermöglichen, ist es notwendig, im Workflow-Modell eine Indirektion zwischen einer logischen und einer physischen Ebene einzuführen. Das Vorbild bildet dabei die aus dem Datenbankbereich stammende Trennung zwischen logischen und physischem Datenmodell [LaLo95]. Die logische Workflow-Modell-Ebene stellt Schemaelemente für eine ressourcenunabhängige Darstellung von Workflow-Schemata bereit. So wird verhindert, daß informationstechnische Ressourcen und damit Implementierungsdetails im Schema erscheinen. Die physische Workflow-Modell-Ebene beschreibt die für die Erfüllung der Anforderungen der logischen Ebene erforderlichen Ressourcen.

Dargestellt wird diese Gestaltung des Workflow-Modells an Hand des Workflow-Metamodells. Es unterscheidet zwischen einer logischen und einer physischen Ebene. Entsprechend diesen beiden Ebenen werden die Aspekte der logischen oder der physischen Ebene zugewiesen. Aspekte der logischen Ebene dienen der Darstellung des Workflows unabhängig von den zur Umsetzung verwendeten Ressourcen. Aspekte der physischen Ebene beschreiben informationstechnische Ressourcen zur Umsetzung von Aspekten der logischen Ebene. Dargestellt ist dies in Abbildung 38.

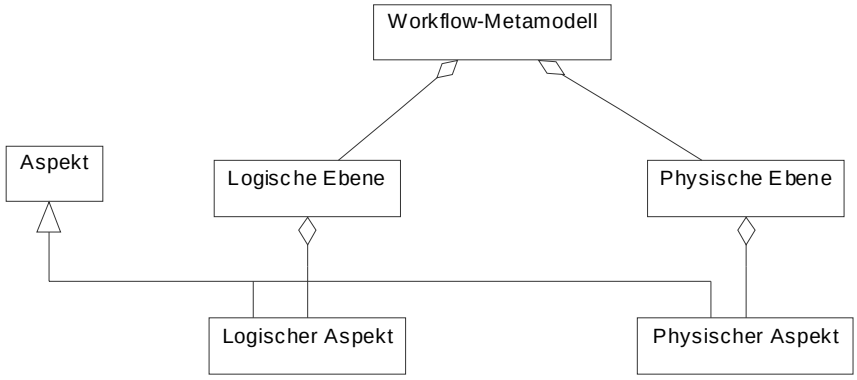


Abbildung 38: Logische und physische Ebene im Zwei-Ebenen-Workflow-Metamodell

Die Zuweisung zwischen physischer und logischer Ebene mit ganzen Aspekten als Granulat reicht jedoch für die angestrebte feingranulare Zuweisung von informationstechnischen Ressourcen nicht aus. So können verschiedene Ressourcen zur Unterstützung eines logischen Aspektes eingesetzt werden. Beispielsweise kann es erforderlich sein, unterschiedliche Datenbanken im Lauf des Workflows einzusetzen. Daher muß eine feinere Zerlegung als auf Aspektebene stattfinden. Logische Aspekte werden in logische Elemente zerlegt, physische Aspekte in physische Elemente. Die Zuweisung zwischen der physischen und der logischen Ebene muß auf der Ebene dieser Aspektelemente erfolgen, um die angestrebte feingranulare Zuweisung zu erreichen. Jedem Element eines logischen Aspekts wird also ein Element eines physischen Aspekts zugewiesen. Mit dieser feingranularen Gestaltung werden zwei Zwecke verfolgt. Erstens wird die feingranulare Zuweisung informationstechnischer Mittel zu Elementen der logischen Ebene möglich. Zweitens wird die Grundlage für die feingranulare Erweiterbarkeit des Workflow-Modells gelegt. Nur wenn bereits auf der Ebene des Workflow-Metamodells ein für die Zwecke der Erweiterbarkeit ausreichend kleines Granulat vorhanden ist, kann dieses bei der Umsetzung unterstützt werden.

In Abbildung 39 ist das auf Basis dieser Überlegungen geschaffene Workflow-Metamodell dargestellt. Es setzt sich aus einer logischen und einer physischen Ebene zusammen, denen jeweils logische und physische Aspekte zugeordnet sind. Diese werden weiter in logische und

physische Elemente zerlegt. Zwischen ihnen findet dann die Verbindung von logischer und physischer Ebene statt.

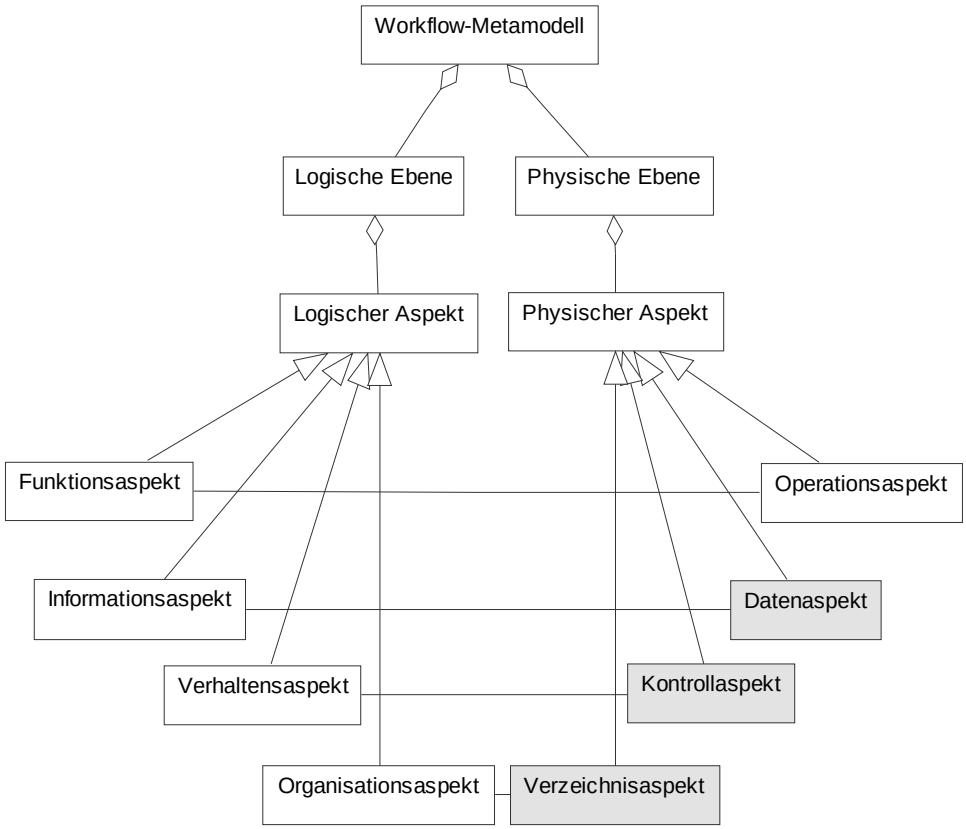


Abbildung 39: Logische und physische Elemente im Zwei-Ebenen-Workflow-Metamodell

6.2.2 Dienste und Ressourcen

Jedes physische Element beschreibt eine Menge von Diensten, die notwendig sind, um das logische Element zu unterstützen, wie Abbildung 40 zeigt. Andererseits kann ein Dienst auch mehrere unterschiedliche physische Elemente unterstützen. Die Dienste werden durch eine oder mehrere Ressourcen bereitgestellt. Diese befinden sich aber nicht innerhalb des Modells, sondern werden durch eine Ressourcenbeschreibung vertreten.

Dienste stellen eine Indirektionsebene zwischen den physischen Elementen und den sie erbringenden Ressourcen dar. Wird ein Dienst durch mehrere Ressourcen bereitgestellt, so wird ein Auswahlmechanismus notwendig. Die dafür notwendigen Konzepte sollen allerdings hier nicht ausgeführt werden.

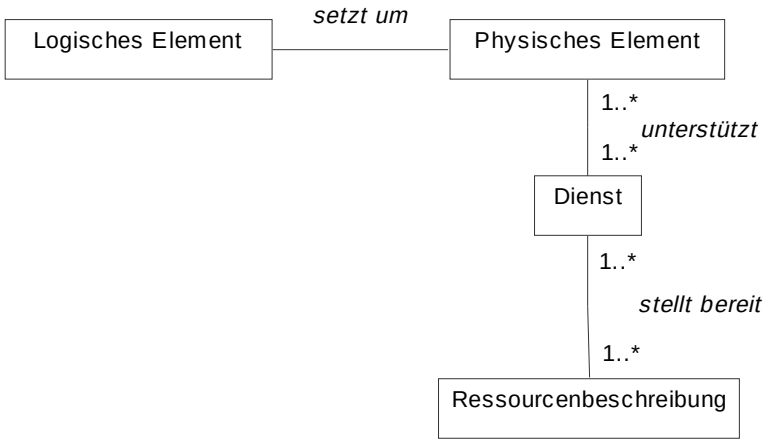


Abbildung 40: Physische Elemente, Dienste und Ressourcenbeschreibungen

Ein Workflow-Modell, das auf der Grundlage des Zwei-Ebenen-Workflow-Metamodells geschaffen wurde, wird als Zwei-Ebenen-Workflow-Modell bezeichnet. Es hat ebenfalls eine logische und eine physische Ebene. Die Elemente der logischen Ebene sind logischen Aspekten zugehörig, die der physischen Ebene physischen Aspekten. Es wird folgendermaßen eingesetzt: Die logische Ebene stellt Schemaelemente zur Umsetzung des Geschäftsprozeßschemas bereit. Mit den Elementen der physischen Ebene werden Dienste und Ressourcen beschrieben. In Form des Zwei-Ebenen-Workflow-Modells werden beide Ebenen zusammengeführt. Dieser Zusammenhang ist in Abbildung 41 dargestellt.

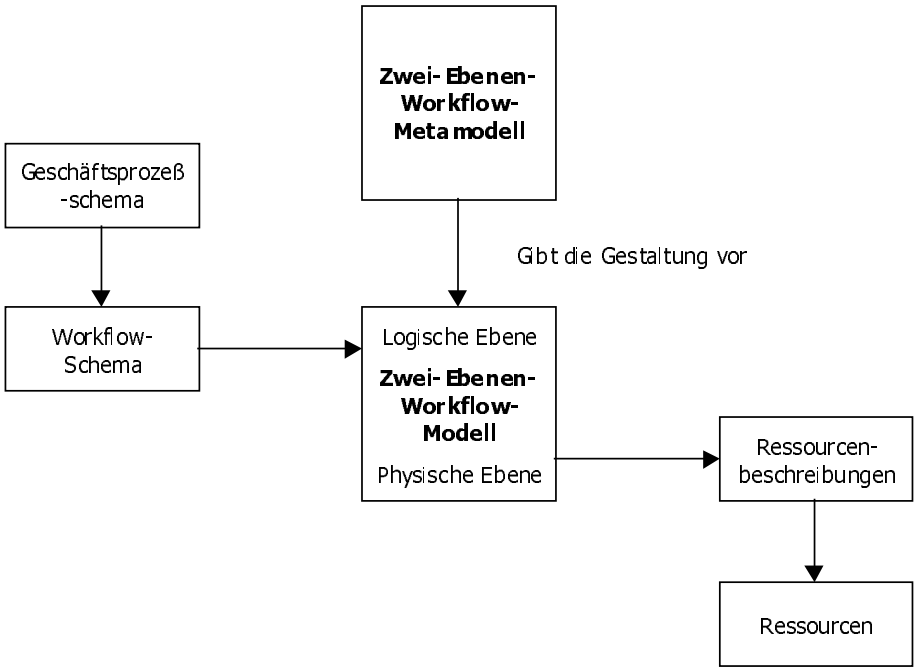


Abbildung 41: Anwendung des Zwei-Ebenen-Workflow-Metamodells

6.2.3 Zusätzliche Aspekte

In diesem Abschnitt sollen die aus Kapitel 2 bekannten Aspekte daraufhin untersucht werden, ob und wie sie sich in das Zwei-Ebenen Workflow-Metamodell einfügen lassen. Falls ein Aspekt der logischen oder physischen Ebene zuordenbar ist, wird geprüft, ob ein dementsprechender physischer bzw. logischer Aspekt als „Spiegelbild“ existiert. Ist dies nicht der Fall, muß ein entsprechender Aspekt neu geschaffen werden. Sollte ein Aspekt beiden Ebenen zuzuordnen sein, so muß eine Aufteilung in zwei neue Aspekte erfolgen. Die Ergebnisse dieser Untersu-

chung sind in Abbildung 42 wiedergegeben. Neu hinzugekommene Aspekte sind grau hinterlegt. Jeweils zusammengehörige Aspekte sind durch eine Assoziation verbunden.

Der Informationsaspekt beschreibt bisher nur das logische Datenmodell der verwendeten Daten, nicht jedoch das dazugehörige physische Datenmodell. Daher ist der Informationsaspekt der logischen Ebene zuzuordnen. Ihm ist auf der physischen Ebene ein Aspekt entgegenzusetzen, der die Darstellung des physischen Datenmodells der Daten übernimmt, er wird als Datenaspekt bezeichnet.

Der Verhaltensaspekt ist der logischen Ebene zuzuordnen. Er wird durch den neu geschaffenen Kontrollaspekt auf der physischen Ebene ergänzt, der die Umsetzung der Elemente des Verhaltensaspekts beschreibt. Die Trennung zwischen Verhaltens- und Kontrollaspekt eröffnet die Möglichkeit, unterschiedliche Mechanismen für die Unterstützung des Verhaltensaspekts einzusetzen. Bisher war die Umsetzung dieses Aspekts fest zugewiesen.

Der Organisationsaspekt beschäftigt sich mit der Darstellung von Personen, deren Rollen und ihre Einbindung in eine Hierarchie. Die informationstechnischen Ressourcen hierzu wurden nicht betrachtet. Der Organisationsaspekt ist daher der logischen Ebene zuzuordnen und auf der physischen Ebene durch einen zusätzlichen Aspekt, den Verzeichnisaspekt zu ergänzen. Aufgabe des Verzeichnisaspekts ist die Beschreibung der informationstechnischen Ressourcen zur Darstellung des Organisationsaspekts.

Die einzigen Aspekte des bisherigen Metamodells, die in dieses Metamodell übertragbar sind, sind der Funktions- und der Operationsaspekt. Der Funktionsaspekt ist der logischen Ebene und der Operationsaspekt der physischen Ebene zuzuordnen. Für die übrigen Aspekte des in Abschnitt 2.3.1 vorgestellten Workflow-Metamodells muß zunächst geklärt werden, welcher Ebene sie zuzuordnen sind und ob zusätzliche Aspekte ergänzt werden müssen.

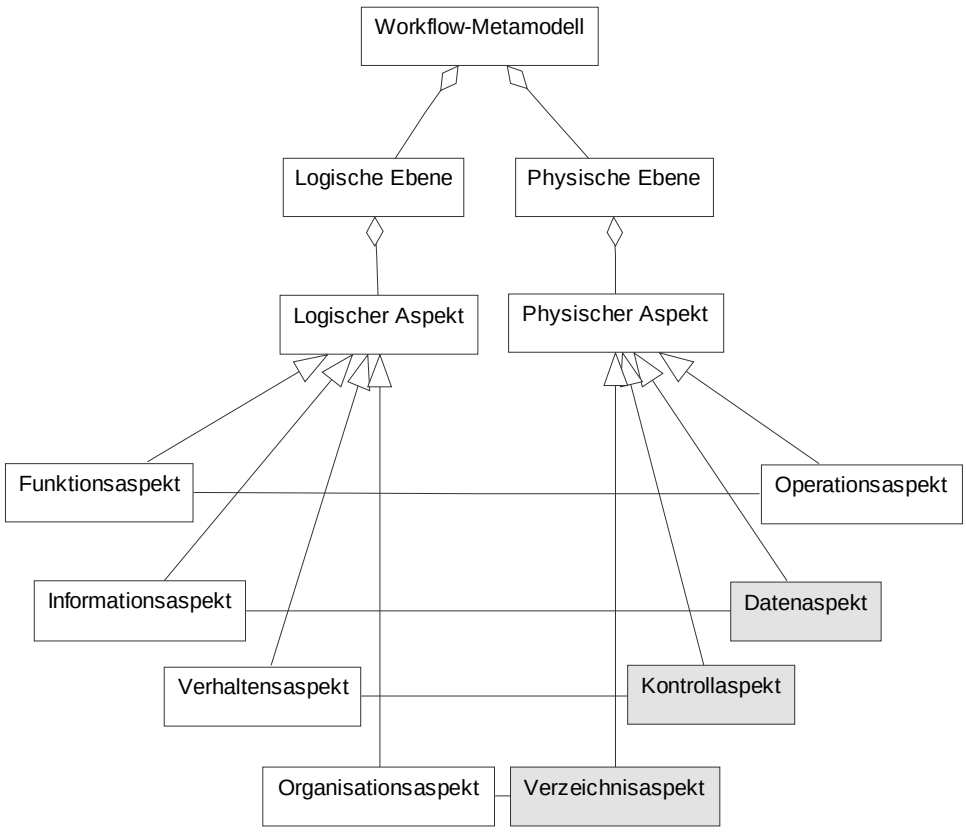


Abbildung 42: Zusätzliche Aspekte im Zwei-Ebenen-Workflow-Metamodell

Die Anwendung des Zwei-Ebenen-Workflow-Metamodells wird an Hand eines Beispiels veranschaulicht. Dazu wird auf der Basis eines Workflow-Modells, das nach dem alten Workflow-Metamodell gestaltet ist, Schritt für Schritt ein Workflow-Modell nach dem Zwei-Ebenen-Workflow-Metamodell geschaffen. In Abbildung 43 ist zunächst das ursprüngliche Workflow-Modell abgebildet.

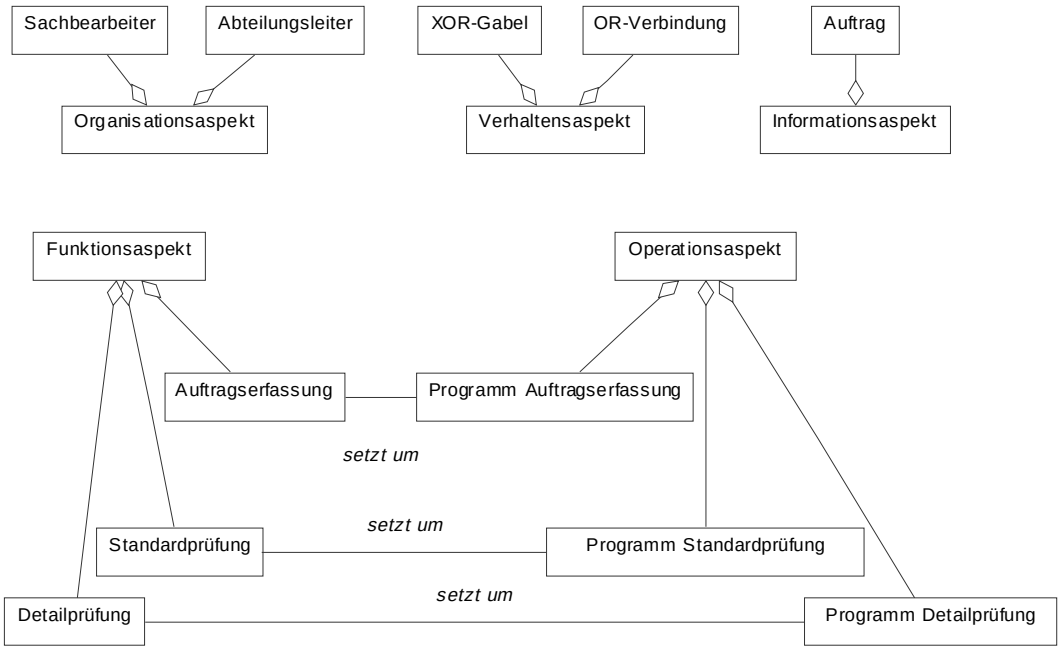


Abbildung 43: Workflow-Modell nach dem alten Workflow-Metamodell

Die Modellelemente werden nun in ein Zwei-Ebenen-Workflow-Modell eingeordnet. Es ist wieder zu beachten, daß nicht Schemaelemente, sondern Modellelemente betrachtet werden. Dabei wird die auf S. 27 eingeführte Notation verwendet, die die Aspektzugehörigkeit von Modellelementen aus Gründen der Übersichtlichkeit als Aggregation darstellt.

6.2.3.1 Informationsaspekt

Die Untersuchung des alten Workflow-Modells auf Seite 71 hat gezeigt, daß der Informationsaspekt durch den Datenaspekt ergänzt werden muß. Dementsprechend müssen auch die Modellelemente des Informationsaspektes durch Elemente des Datenaspekts ergänzt werden. So wird die Speicherungsstruktur für das Element „Auftrag“ des Informationsaspekts durch das Element „D Auftrag“ angegeben, dargestellt in Abbildung 44. Der Buchstabe D gibt die Zugehörigkeit zum Datenaspekt wieder.

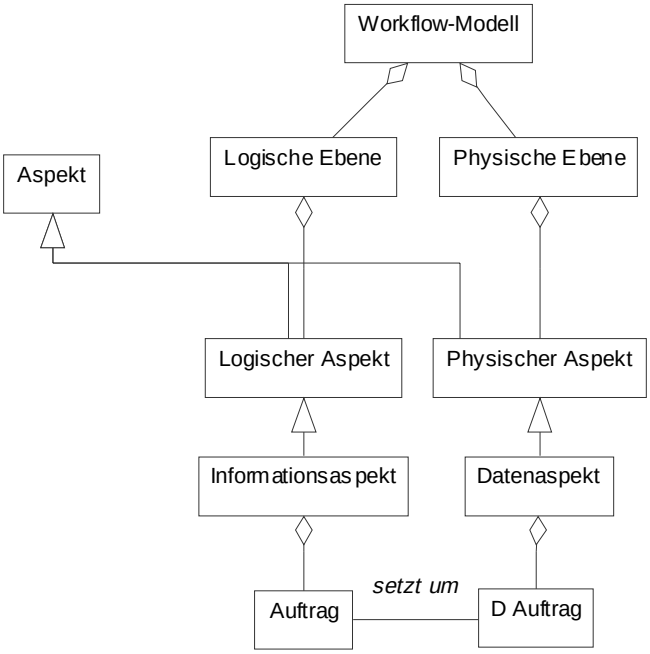


Abbildung 44: Informations- und Datenaspekt

Das physische Element „D Auftrag“ besteht aus der Beschreibung des Dienstes „D D Auftrag“, zu dessen Bereitstellung die Ressource „R D Auftrag“ verwendet wird. Dargestellt ist dies in Abbildung 45.

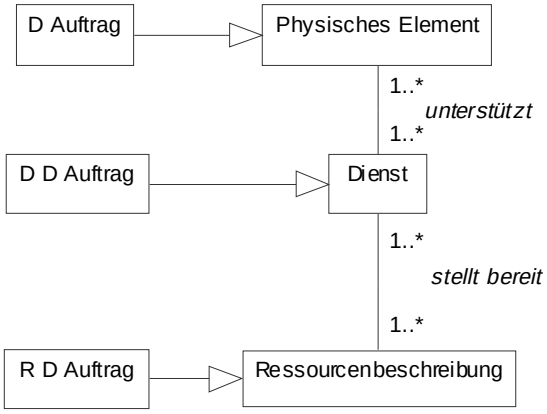


Abbildung 45: Dienste und Ressourcen für D Auftrag

6.2.3.2 Verhaltensaspekt

Auch die Elemente des Verhaltensaspekts werden durch Elemente des Kontrollaspekts ergänzt, wie Abbildung 46 zeigt. Die Elemente des Kontrollaspektes werden durch ein vorangestelltes K kenntlich gemacht, also "K XOR-Gabel" und "K OR-Verbindung"

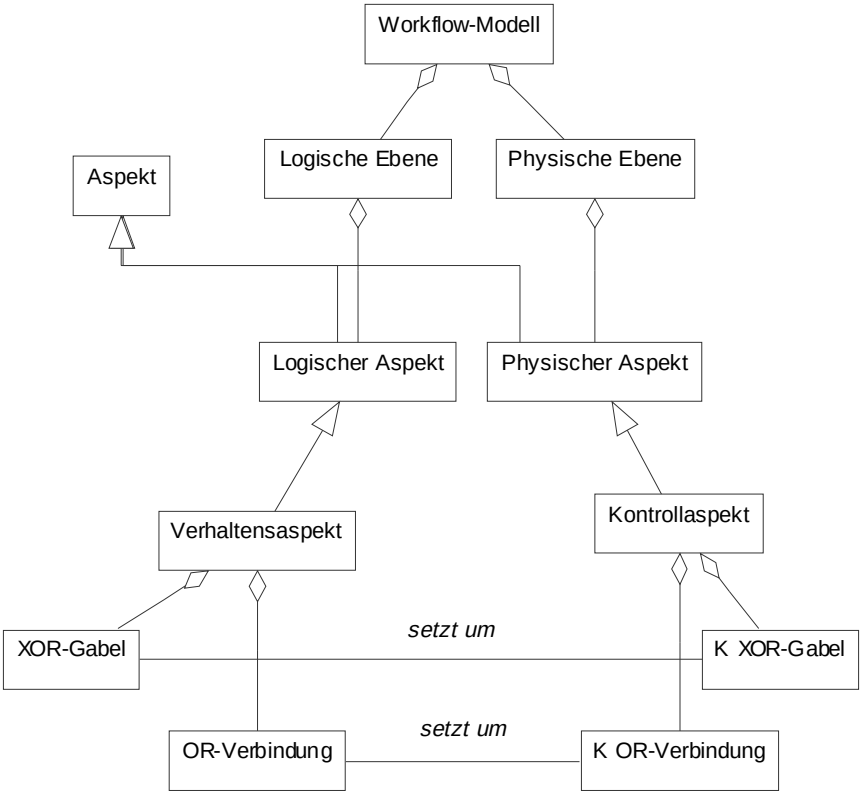


Abbildung 46: Verhaltens- und Kontrollaspektelemente

Das physische Element „K XOR-Gabel“ wird näher betrachtet. Seine Unterstützung geschieht durch den Dienst „D XOR-Gabel“. Dieser Dienst wird wiederum durch die Ressource „R XOR-Gabel“ erbracht. Veranschaulicht ist dies in Abbildung 47.

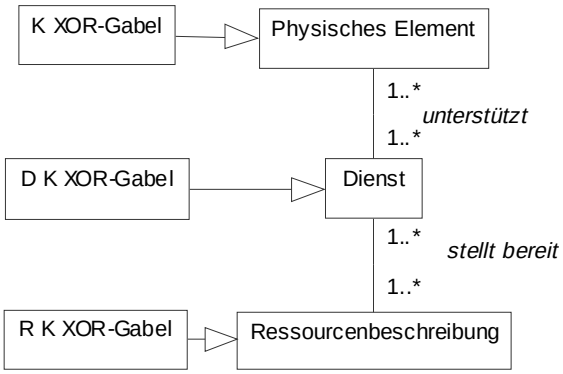


Abbildung 47: Dienste und Ressourcen für K XOR Gabel

6.2.3.3 Organisationsaspekt

Die Elemente des Organisationsaspekts, die für reale Personen, Rollen oder organisatorische Einheiten stehen, werden durch Elemente des Verzeichnisaspekts ergänzt, wie in Abbildung 48 dargestellt. Dem logischen Element Abteilungsleiter ist dementsprechend ein physisches Element "V Abteilungsleiter" zugeordnet. Gleiches gilt für das logische Element Sachbearbeiter, dem das physische Element "V Sachbearbeiter" zugewiesen ist.

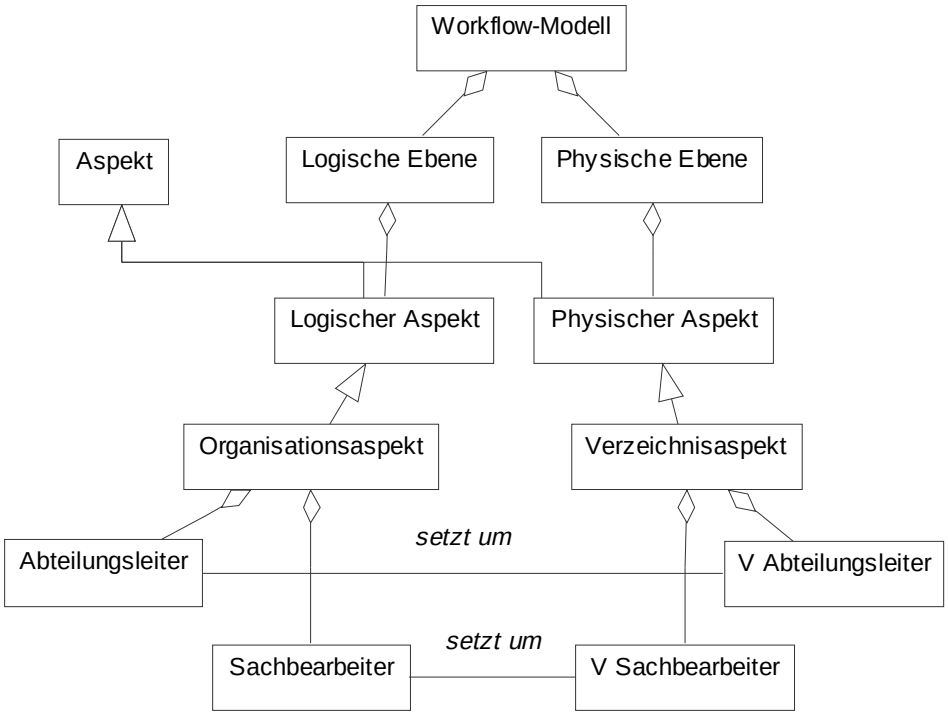


Abbildung 48: Organisations- und Verzeichnisaspektelemente

Das physische Element „V Abteilungsleiter“ wird näher betrachtet. Es wird durch den Dienst „D V Abteilungsleiter“ unterstützt der wiederum durch die Ressource „R V Abteilungsleiter“ erbracht wird. Dies ist in Abbildung 49 wiedergegeben.

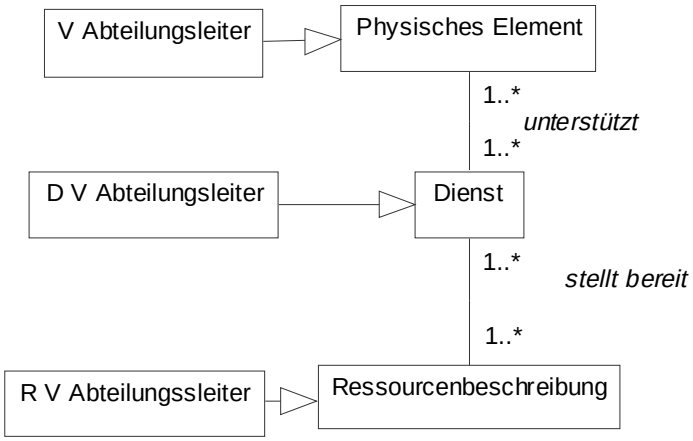


Abbildung 49: Dienste und Ressourcen für V Abteilungsleiter

6.2.3.4 Funktions- und Operationsaspekt

Die Elemente des Funktions- und Operationsaspekts können übernommen werden, wie Abbildung 50 zeigt. Allerdings wird durch das Zwei-Ebenen-Workflow-Metamodell eine feingranularere Darstellung innerhalb des Workflow-Modells bedingt. Dem logischen Element Auftragserfassung ist das physische Element "Programm Auftragserfassung" zugeordnet. Gleiches gilt für die Paare Standardprüfung und "Programm Standardprüfung" sowie "Detailprüfung" und "Programm Detailprüfung".

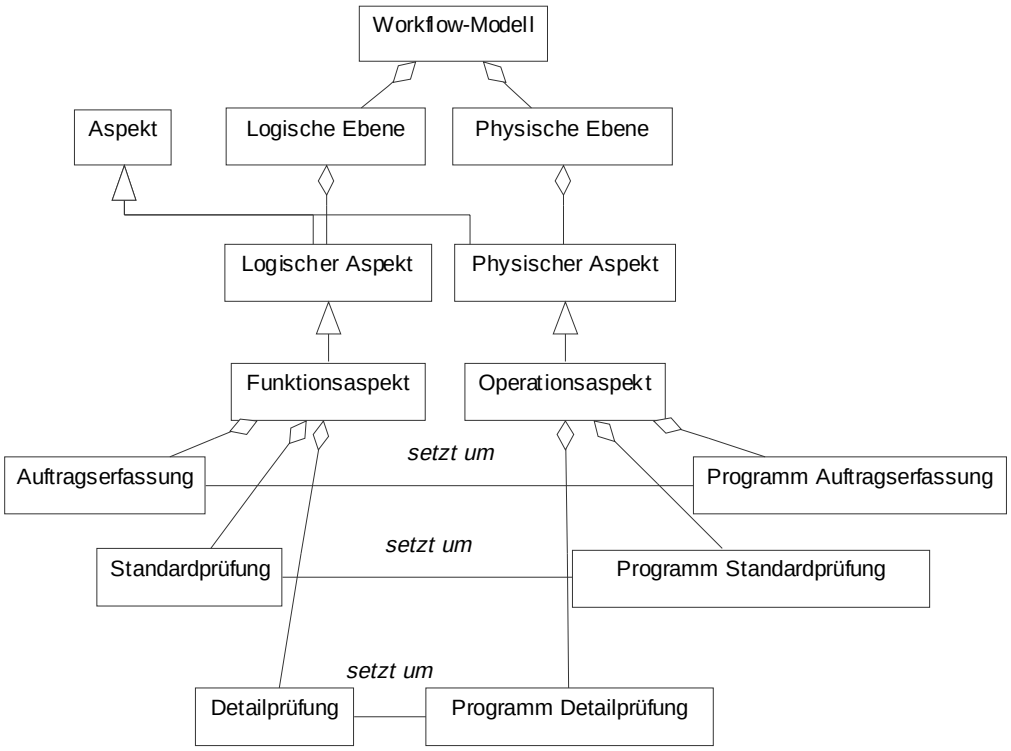


Abbildung 50: Funktions- und Operationsaspekt

Für das physische Element „Programm Auftragserfassung“ wird eine detailliertere Darstellung durchgeführt werden. Es wird durch den Dienst „D Programm Auftragserfassung“ unterstützt, welcher wiederum durch die Ressource „R Programm Auftragserfassung“ erbracht wird. Dieser Zusammenhang ist in Abbildung 51 dargestellt.

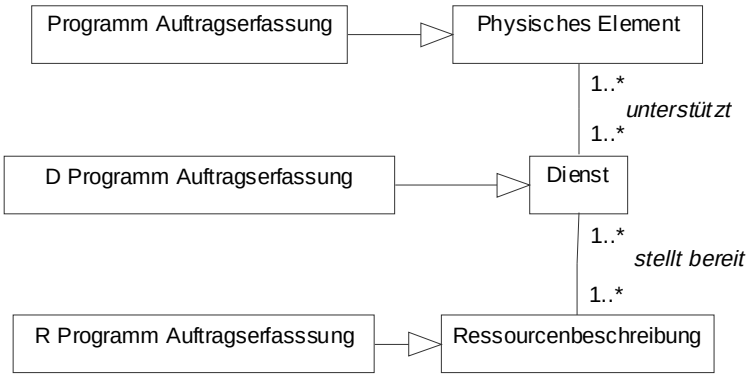


Abbildung 51: Dienste und Ressourcen für Programm Auftragserfassung

Wird das in Abbildung 43 dargestellte Beispielschema dem in Abbildung 42 dargestellten Zwei-Ebenen-Workflow-Modell entsprechend dargestellt, so erhält man das in Abbildung 52 dargestellte Schema. Der entscheidende Unterschied liegt im Wegfallen von Informationen bezüglich informationstechnischer Ressourcen, die in der physischen Ebene des Workflow-Modells abgelegt sind.

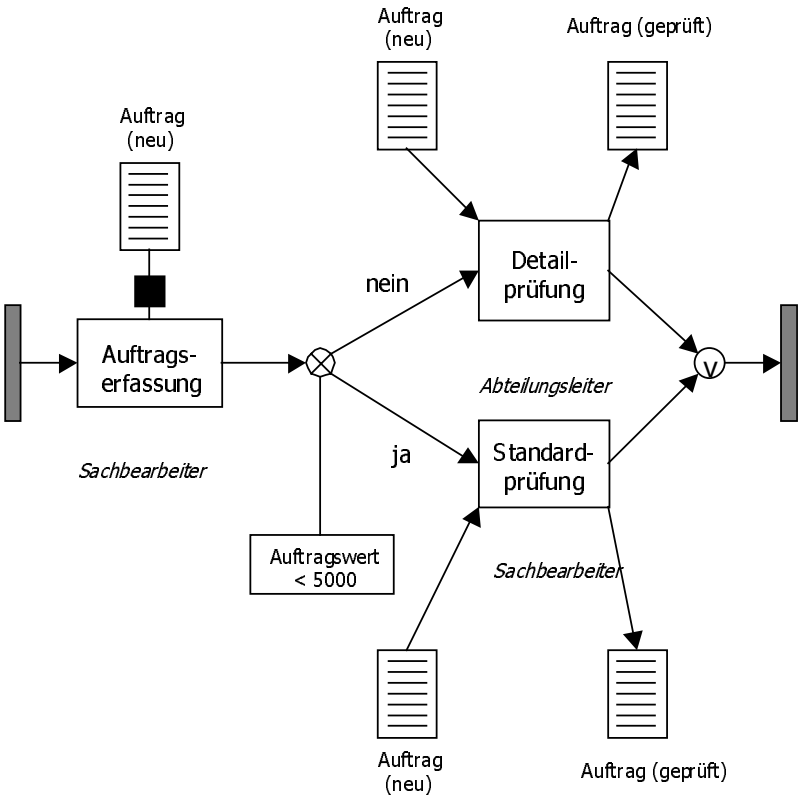


Abbildung 52: Logische Ebene eines Schemas nach dem Zwei-Ebenen-Workflow-Metamodell

6.3 Bewertung

An Hand eines Beispieles wird untersucht, ob das entwickelte Workflow-Metamodell die Grundlage dafür legt, daß von ihm abgeleitete Workflow-Modelle zur dynamischen und feingranularen Zuweisung von Ressourcen in der Lage sind. Zur Veranschaulichung wird der Fall betrachtet, daß die Ressource für die Erbringung des Dienstes „D Programm Auftragserfassung“ und damit die Unterstützung des physischen Elementes „Programm Auftragserfassung“ nicht mehr durch die Ressource „R Programm Auftragserfassung“ sondern durch die Ressource „R Programm Auftragserfassung 2“ erbracht werden soll. Dies kann in einem Workflow-Modell, das auf der Basis des hier entwickelten Zwei-Ebenen-Workflow-Metamodells entwickelt worden ist, einfach wiedergegeben werden, indem die Ressourcenbeschreibung ausgetauscht wird. Für das gewählte Beispiel ist das Resultat in Abbildung 53 abgebildet.

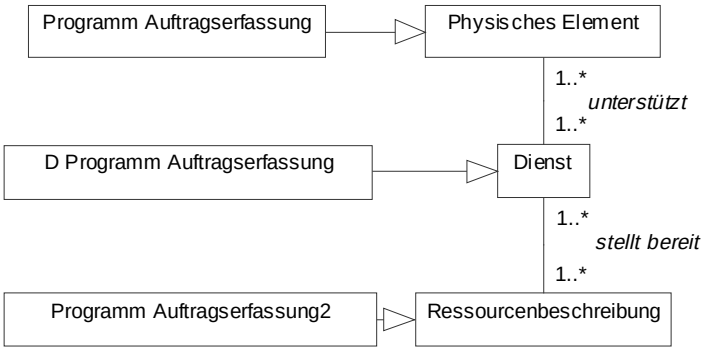


Abbildung 53: Feingranulare Zuweisung von Ressourcen

Es kann also davon ausgegangen werden, daß das hier entwickelte Zwei-Ebenen-Workflow-Metamodell die gestellte Anforderung der dynamischen und feingranularen Zuweisung informationstechnischer Ressourcen erfüllt. Nun wird untersucht, ob und wenn ja welche nicht-funktionale Anforderungen sich aus ihm ableiten lassen.

6.4 Nicht-funktionale Anforderungen

Die im Zwei-Ebenen-Workflow-Metamodell geschaffene Möglichkeit zur dynamischen und feingranularen Zuweisung von Ressourcen bezieht sich genau genommen auf Ressourcenbeschreibungen. Die eigentlichen Ressourcen befinden sich außerhalb des im folgenden Kapitel geschaffenen Workflow-Modell-Wörterbuchs. Damit aber die auf Modellebene vorgenommene Zuweisung von Ressourcen auch realisiert werden kann, ist es notwendig, daß eine umsetzende Software-Architektur die Anforderung der Ressourcenkonfigurierbarkeit erfüllt. Sie besagt, daß die Bereitstellung von Diensten in einer Software-Architektur durch veränderliche Ressourcen erfolgen können muß. Die Ressourcen, die einen bestimmten Dienst erbringen, müssen zur Laufzeit des Software-Systems auswechselbar sein.

6.5 Zusammenfassung

Ziel dieses Kapitels war, ein Workflow-Metamodell zu schaffen, das die dynamische und feingranulare Zuweisung informationstechnischer Ressourcen für die Prozeßunterstützung ermöglicht.

Dieses Ziel wurde durch das geschaffene Zwei-Ebenen-Workflow-Metamodell erreicht. Durch die Verwendung von zwei Ebenen wird die dynamische Zuweisung von Ressourcen ermöglicht. Die logische Ebene legt die Grundlage für eine von konkreten informationstechnischen Ressourcen unabhängige Darstellung von Workflow-Schemata. Die physische Ebene schafft zu jedem Aspekt der logischen Ebene einen korrespondierenden Aspekt, der die informationstechnischen Ressourcen zur Unterstützung des logischen Aspekts darstellt. Die Zuordnung zwischen logischer und physischer Ebene auf der Basis von Aspektelementen ermöglicht die feingranulare Zuweisung von Ressourcen. Durch die Verwendung von Aspektelementen als Granulat können für einzelne Elemente eines Aspektes unterschiedliche informationstechnische Ressourcen zur Unterstützung zugewiesen werden.

In einem weiteren Schritt wurden dann bereits bekannte Aspekte auf ihre Übertragbarkeit in das Zwei-Ebenen-Workflow-Modell überprüft. Nur der Funktions- und der Operationsaspekt konnten direkt übernommen werden. Für sie mußten keine zusätzlichen Aspekte als Ergänzung auf der logischen bzw. physischen Ebene geschaffen werden. Alle anderen untersuchten Aspekte wurden der logischen Ebene zugeordnet und mußten durch neu geschaffene Aspekte auf der physischen Ebene ergänzt werden.

Als Anforderung an die umsetzende Software-Architektur wurde schließlich die nicht-funktionale Anforderung der Ressourcenkonfigurierbarkeit abgeleitet.

7 Workflow-Modell-Wörterbuch

Ziel dieses Kapitels ist es, die dynamische Repräsentation des Workflow-Modells zu ermöglichen. Auf diese Weise kann die Menge der verarbeitbaren Schemaelemente eines WfMS zur Laufzeit erweitert werden. Hierzu wird zunächst untersucht, ob bereits existierende Ansätze Beiträge zur Lösung dieses Problems liefern können. Dabei stellt sich heraus, daß diese zwar Konzepte zur Erweiterung der Menge der Schemata, nicht aber zur Erweiterung des Workflow-Modells besitzen. Übernommen werden kann jedoch die Verwendung eines Wörterbuchs. Daher wird das Konzept des Workflow-Modell-Wörterbuchs entwickelt. Wichtige Schritte dabei sind die Bestimmung des Informationsmodells des Wörterbuchs und der Operationen auf dem Wörterbuch.

7.1 Existierende Ansätze

Existierende Ansätze, die das Ziel der Erweiterbarkeit des Workflow-Modells verfolgen, sind der Mobile [Jabl94], [JaBS97] und der Mentor-lite-Ansatz [MWGW98]. Mobile hat die modulare und insbesondere erweiterbare Gestaltung von Workflow-Management-Systemen zum Ziel. Die Mobile-Architektur sieht die Schaffung eines Workflow-Systems durch Module vor, die jeweils einen Aspekt in seiner Gesamtheit unterstützen, jedoch keine weitere Zerlegung durchführen. Die Anforderung der dynamischen Erweiterbarkeit erreicht es jedoch nicht, da die Erweiterbarkeit nur während des Entwurfs gegeben ist. Erweiterungen eines Mobile-WfMS erfordern das Durchlaufen des gesamten Entwicklungszyklus, d.h. Quellcodeveränderung, Übersetzung, Installation usw. Der Mentor-lite Ansatz besitzt die Möglichkeit, Erweiterungen als Workflows darzustellen und durch die Workflow-Maschine zu interpretieren. Auf diese Weise können in Mentor-lite zusätzliche Aspekte und Aspektelemente wie das Workflow-Monitoring oder die Workflow-Historie hinzugefügt werden. Die Workflow-Maschine stellt jedoch die einzige Ressource dar, durch die die Erweiterungen unterstützt werden. Daher verletzt der Mentor-lite Ansatz die Anforderung der autarken Zuweisung von Ressourcen. Höchst problematisch ist zudem die Schachtelung von Interpretationsvorgängen. So wird allgemein die zentralisierte und interpretative Ausführung von Workflows als problematisch für die Skalierbarkeit angesehen [JaBS97]. Dies gilt dann um so mehr, wenn zwei Interpretationsvorgänge verschachtelt sind.

Während also die Erweiterbarkeit von Workflow-Modellen in bisherigen Ansätzen nicht unterstützt wird, gibt es einen Bereich, in dem die Erweiterbarkeit erreicht wird. Es ist dies die Erweiterbarkeit um zusätzliche Workflow-Schemata. So setzen eine Reihe von Ansätzen Wörterbücher zur Speicherung der Workflow-Schemata ein und erreichen so die Erweiterbarkeit um zusätzliche Workflow-Schemata, beispielsweise [ObWS94], [BöMS97], [WWDM97], [Riem98], [GeTo98], [SKMW96], [Schu99], [Lehm99]. Daher wird diese Anregung aufgegriffen und ein Wörterbuch für die Repräsentation des Workflow-Modells geschaffen werden.

7.2 Konzeption

Der Unterschied zwischen dem hier verwendeten Workflow-Modell-Wörterbuch und dem der existierenden Ansätze wird an Hand von Abbildung 54 veranschaulicht. Das Workflow-Modell-Wörterbuch repräsentiert ein Workflow-Modell, das auf der Basis des Zwei-Ebenen-Workflow-Metamodells gebildet worden ist. Auf der Basis dieses Workflow-Modells werden wiederum Workflow-Schemata gebildet, die in einem eigenen Wörterbuch abgelegt werden. Die von den existierenden Ansätzen durchgeführte Speicherung von Schemata in einem Workflow-Schema-Wörterbuch kann übernommen werden und wird daher hier nicht weiterverfolgt.

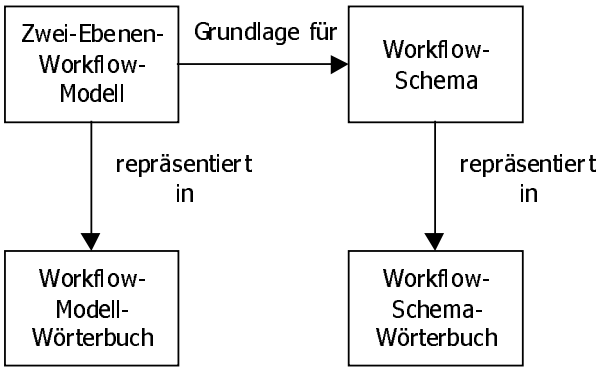


Abbildung 54: Workflow-Modell- und Schema-Wörterbuch

7.2.1 Informationsmodell des Workflow-Modell-Wörterbuchs

Ein Wörterbuch (oder auch Repository) [Bern96], [BeDa94], [Bern98], [BHSS97] ist eine Datenbank, die Informationen über Artefakte, in diesem Fall das Workflow-Modell, enthält. Ein Wörterbuch besteht aus einer Datenbank, einem Wörterbuch-Verwalter und dem Wörterbuch-Informationsmodell[5], wie in Abbildung 55 dargestellt. Das Wörterbuch wird von Werkzeugen zum Einbringen von Daten genutzt.

[5] Das Informationsmodell eines Wörterbuchs ist auch eine Art von Metamodell, allerdings für die im Wörterbuch befindlichen Informationen. Es wäre daher falsch, es mit dem Workflow-Metamodell gleichzusetzen. Um Verwechslungen mit dem Workflow-Metamodell zu vermeiden, soll daher die in der Literatur eingeführte Bezeichnung Informationsmodell beibehalten werden.

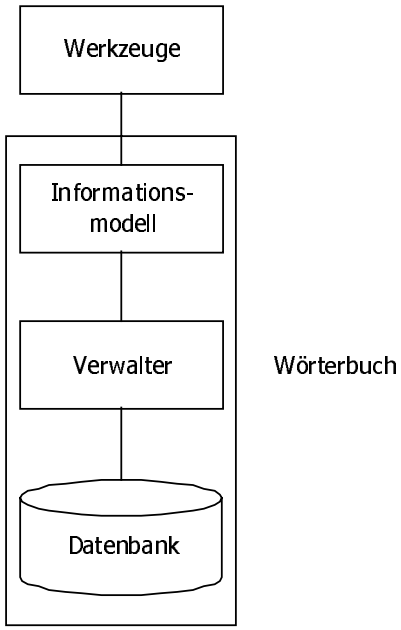


Abbildung 55: Bestandteile eines Wörterbuchs

Das Wörterbuch-Informationsmodell besteht, wie in Abbildung 56 dargestellt, aus typisierten Objekten, die über Beziehungen zueinander in Verbindung stehen (Die Darstellung erfolgt in UML-Notation). Die Aufgabe bei der Entwicklung des Informationsmodells ist es, geeignete Objekttypen und Beziehungstypen zu definieren, auf deren Basis die Darstellung der gewünschten Informationen erfolgen kann.

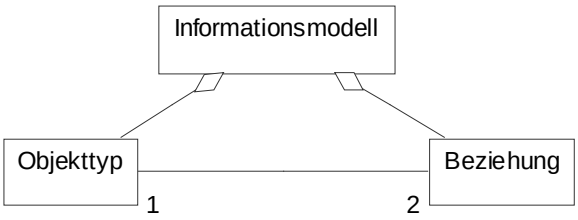


Abbildung 56: Wörterbuch-Informationsmodell

In dieser Arbeit wird nicht auf die Gestaltung der Wörterbuch-Datenbank oder des Wörterbuch-Verwalters eingegangen. Ausführliche Darstellungen dieser Bestandteile finden sich in [BöMS97], [Bern96], [BHSS97] und [OMGM]. Wichtig ist hingegen die Gestaltung des Informationsmodells des Wörterbuchs.

Aufgabe des Informationsmodells des Workflow-Modell-Wörterbuchs ist es, die Struktur und Semantik der in ihm abgelegten Informationen zu beschreiben. Dazu werden Objekttypen und sie verbindende Beziehungen beschrieben. Es gilt, ein solches Informationsmodell für die Speicherung von Workflow-Modellen zu entwickeln. Bei den abgelegten Informationen handelt es sich um das Workflow-Modell, das wiederum durch das in Abschnitt 6 entwickelte Zwei-Ebenen-Workflow-Metamodell beschrieben wird. Daher liegt es nahe, das Workflow-Metamodell als Grundlage des Informationsmodells des Workflow-Modell-Wörterbuchs zu verwenden. Veranschaulicht ist dieser Zusammenhang in Abbildung 57. Das Zwei-Ebenen-Workflow-Metamodell gibt also die Struktur des Informationsmodells vor, ist aber mit diesem nicht identisch. Durch das Informationsmodell werden die Repräsentationsstrukturen der Datenbank des Workflow-Modell-Wörterbuchs festgelegt, mit der das Workflow-Modell repräsentiert wird.

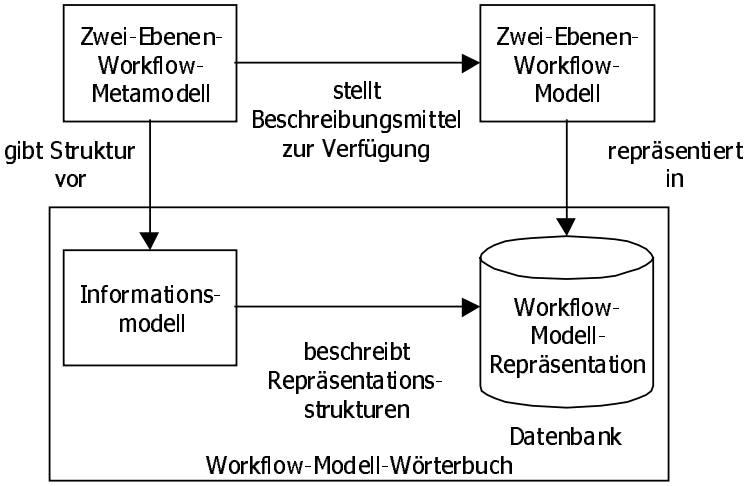


Abbildung 57: Konzepte im Workflow-Modell-Wörterbuch

Das Informationsmodell des Workflow-Modell-Wörterbuchs kennt dementsprechend sechs Objekttypen: Logische und physische Aspekte, logische und physische Elemente, Dienste und Ressourcenbeschreibungen. Aus Gründen der Übersichtlichkeit wird es in zwei Teilen dargestellt. Der erste Teil des Informationsmodells ist in Abbildung 58 dargestellt.

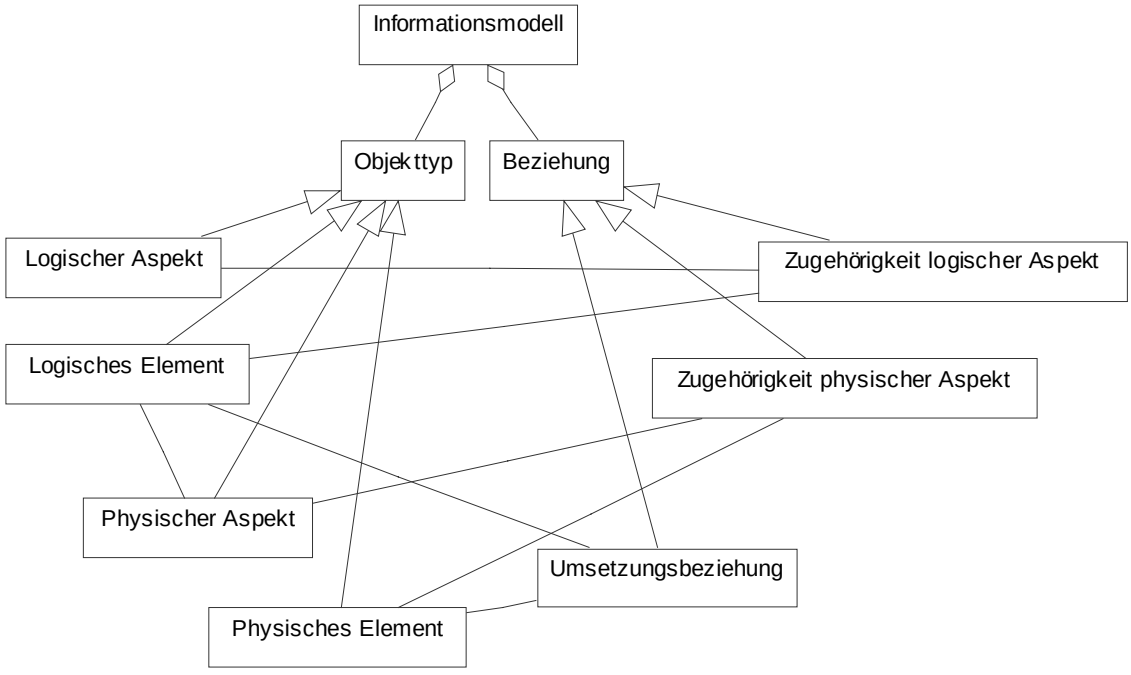


Abbildung 58: Informationsmodell des Workflow-Modell-Wörterbuchs (1)

In ihm sind die Objekttypen „Logischer Aspekt“, „Logisches Element“, „Physischer Aspekt“ und „Physisches Element“ wiedergegeben. Zwischen den Objekttypen „Logischer Aspekt“ und „Logisches Element“ besteht die Beziehung „Zugehörigkeit logischer Aspekt“. Zwischen den Objekttypen„Physischer Aspekt“ und „Physisches Element“ besteht die Beziehung „Zugehörigkeit physischer Aspekt“. Zwischen den Objekttypen „Logisches Element“ und „Physisches Element“ besteht die Umsetzungsbeziehung.

Der zweite Teil des Informationsmodells beschreibt die Objekttypen „Physisches Element“, „Dienstbeschreibung“ und „Ressourcenbeschreibung“. Es ist in Abbildung 59 wiedergegeben. Zwischen den Objekttypen des zweiten Teils des Informationsmodells bestehen zwei Beziehungen: Zwischen den Objekttypen „Physisches Element“ und „Dienstbeschreibung“ besteht die

Unterstützungsbeziehung. Zwischen den Objekttypen „Dienstbeschreibung“ und „Ressourcenbeschreibung“ besteht die Bereitstellungsbeziehung.

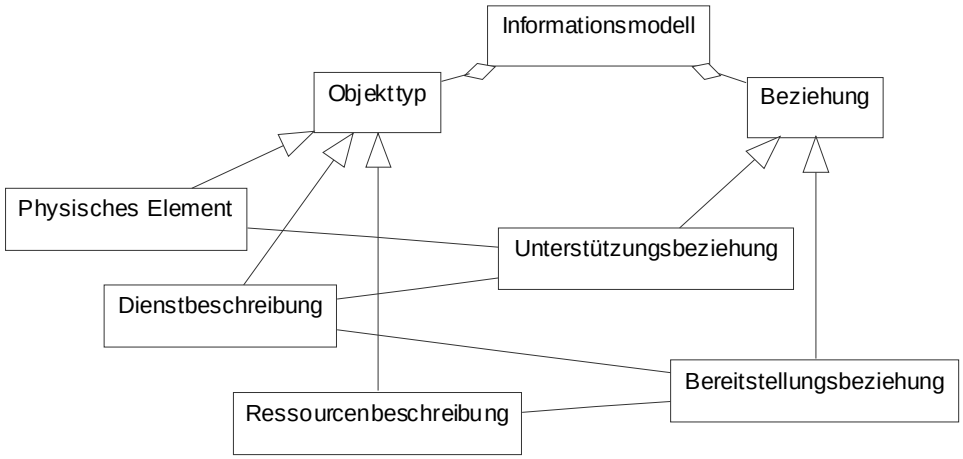


Abbildung 59: Informationsmodell des Workflow-Modell-Wörterbuchs (2)

7.2.2 Operationen auf dem Workflow-Modell-Wörterbuch

Um die geforderte Erweiterbarkeit und Konfigurierbarkeit des Workflow-Modells sicherzustellen, muß das Workflow-Modell-Wörterbuch eine Reihe von Operationen unterstützen, die im folgenden beschrieben werden sollen.

Um Erweiterungen des Workflow-Modells um zusätzlicher Aspekte repräsentieren zu können, muß es möglich sein, die Objekttypen „Logischer Aspekt“ und „Physischer Aspekt“ zu erzeugen. Für die Aufnahme zusätzlicher Aspektelemente müssen die Objekttypen „Logisches Element“ und „Physisches Element“ erzeugt werden können. Zur Repräsentation der Erweiterung des Workflow-Modells um zusätzliche Ressourcen müssen die Objekttypen „Dienstbeschreibung“ und „Ressourcenbeschreibung“ erzeugt werden können. Konsistenzbedingung dabei ist, daß nicht schon ein Objekttyp gleichen Namens existiert. Ebenso muß es möglich sein, die oben genannten Objekttypen zu löschen. Dabei ist allerdings darauf zu achten, daß diese Objekttypen nicht mehr in Verwendung sind. Logische Aspekte und deren Elemente können gelöscht werden, wenn es kein Workflow-Schema mehr gibt, das diese Elemente verwendet. Physische Elemente können gelöscht werden, wenn sich kein Schema mehr in der Ausführung befindet, das diese Elemente benutzt.

Die Unterschiede für die möglichen Löschzeitpunkte für physische und logische Elemente mag zunächst befremden. Ruft man sich jedoch die indirekte Bildung von Ausprägungen ins Gedächtnis, so sieht man, daß die logischen und physischen Elemente zu unterschiedlichen Zeitpunkten benötigt werden. Daher kann bei indirekt Ausprägungen bildenden Ansätzen die Löschung getrennt erfolgen, bei direkt Ausprägungen bildenden Ansätzen muß sie gleichzeitig erfolgen. Ebenso müssen Beziehungen der Typen „Zugehörigkeit logischer Aspekt“, „Zugehörigkeit physischer Aspekt“ und die Umsetzungsbeziehung erstellt und gelöscht werden können. Dies ist in Abbildung 60 dargestellt.

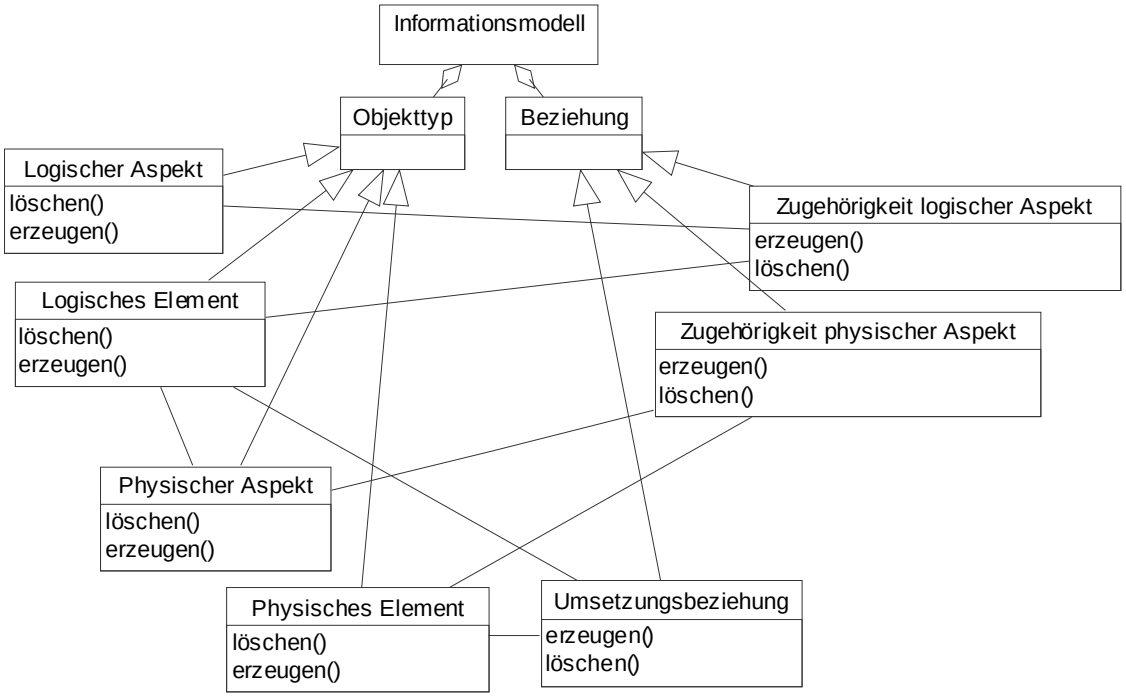


Abbildung 60: Operationen auf dem Informationsmodell (1)

Für die dynamische Zuweisung von informationstechnischen Ressourcen muß es möglich sein, die Unterstützungsbeziehung und die Bereitstellungsbeziehung zu erstellen und zu löschen. Hierfür stehen spezielle Erzeugen-Operationen zur Verfügung, die die zu verknüpfenden Objekttypen als Parameter haben. Sie sind in Abbildung 61 wiedergegeben. Die Erzeugung der Unterstützungsbeziehung hat ein physisches Element und eine Dienstbeschreibung als Parameter. Die Erzeugung der Bereitstellungsbeziehung hat eine Dienstbeschreibung und eine Ressourcenbeschreibung als Parameter.

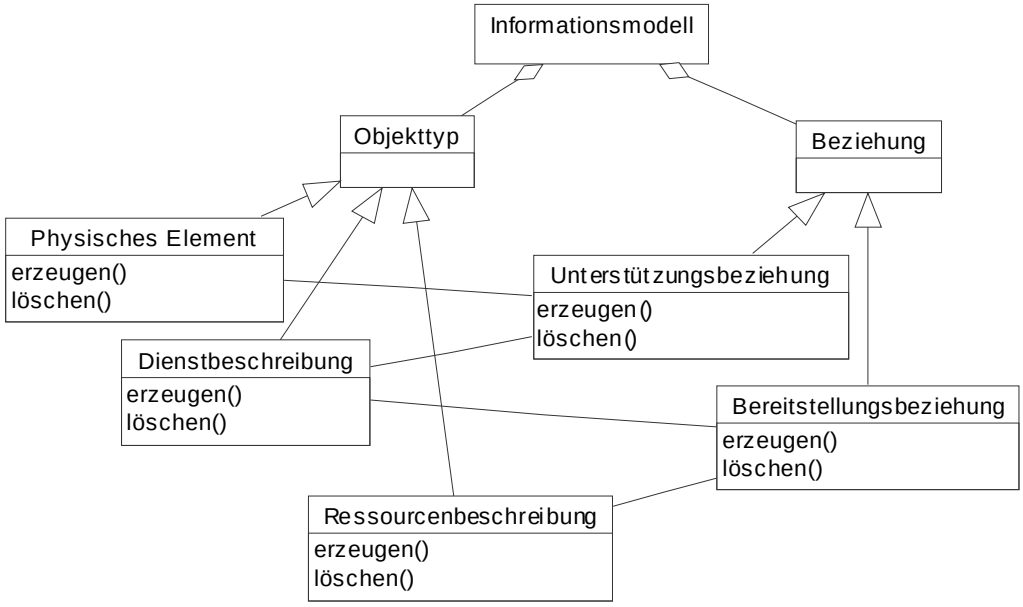


Abbildung 61: Operationen auf dem Informationsmodell (2)

7.3 Bewertung

In diesem Abschnitt wird untersucht, ob das angestrebte Ziel der Schaffung einer dynamischen Modellrepräsentation durch das konzipierte Workflow-Modell-Wörterbuch verwirklicht werden

kann. Dazu muß überprüft werden, ob es möglich ist, in das Workflow-Modell-Wörterbuch zur Laufzeit zusätzliche Elemente aufzunehmen. Dies ist Voraussetzung für die Verarbeitung von Workflow-Schemata, die diese bisher unbekannten Aspektelemente verwenden.

Zur Veranschaulichung dieser Untersuchung wird die bereits früher verwendete Einführung einer Embargoprüfung verwendet. Um dieses erweiterte Workflow-Schema bilden zu können, muß daher eine dementsprechende Erweiterung des Workflow-Modells vorgenommen werden. Die Erweiterung betrifft den Funktions- und Operationsaspekt. In Abbildung 62 ist das zusätzliche Element des Funktionsaspekts, "Embargoprüfung", sowie das zusätzliche Element des Operationsaspekts "Programm Embargoprüfung" eingetragen. Nicht dargestellt sind der Dienst, der die Durchführung der Embargoprüfung bezeichnet, und die Ressource, die diesen Dienst erbringt.

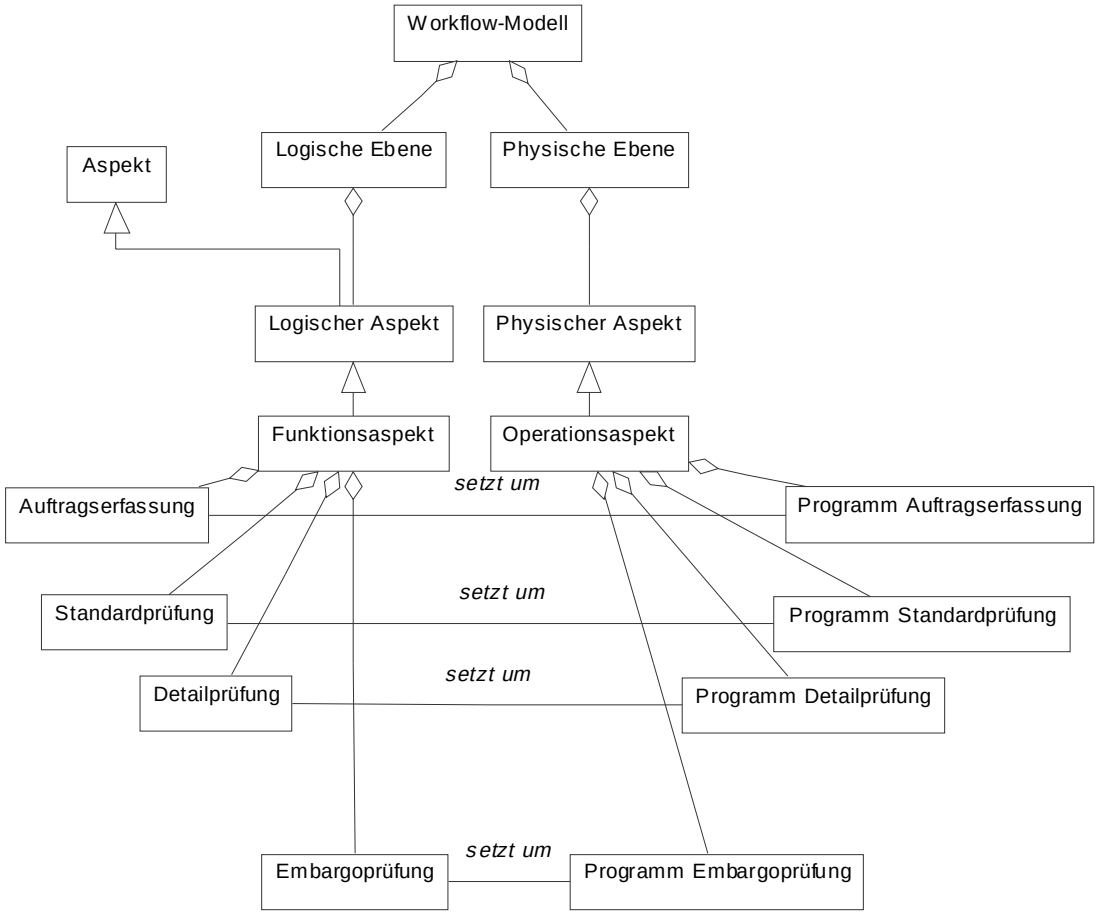


Abbildung 62: Erweiterungen des Workflow-Modells

Im Workflow-Modell-Wörterbuch schlägt sich diese Erweiterung folgendermaßen nieder. Zunächst wird ein zusätzliches logisches Modellelement „Embargoprüfung“ eingeführt, wie in Abbildung 63 veranschaulicht. Ergänzt wird dieses logische Element durch ein physisches Element „Programm Embargoprüfung“ genannt. Unterstützt wird dieses durch den Dienst „Embargoprüfung durchführen“. Dieser wird von der Ressource „Ressource Außenhandelsprüfungen“ bereitgestellt. Letztere stellt noch weitere Dienste für den Außenhandel bereit, die aber hier nicht dargestellt werden sollen.

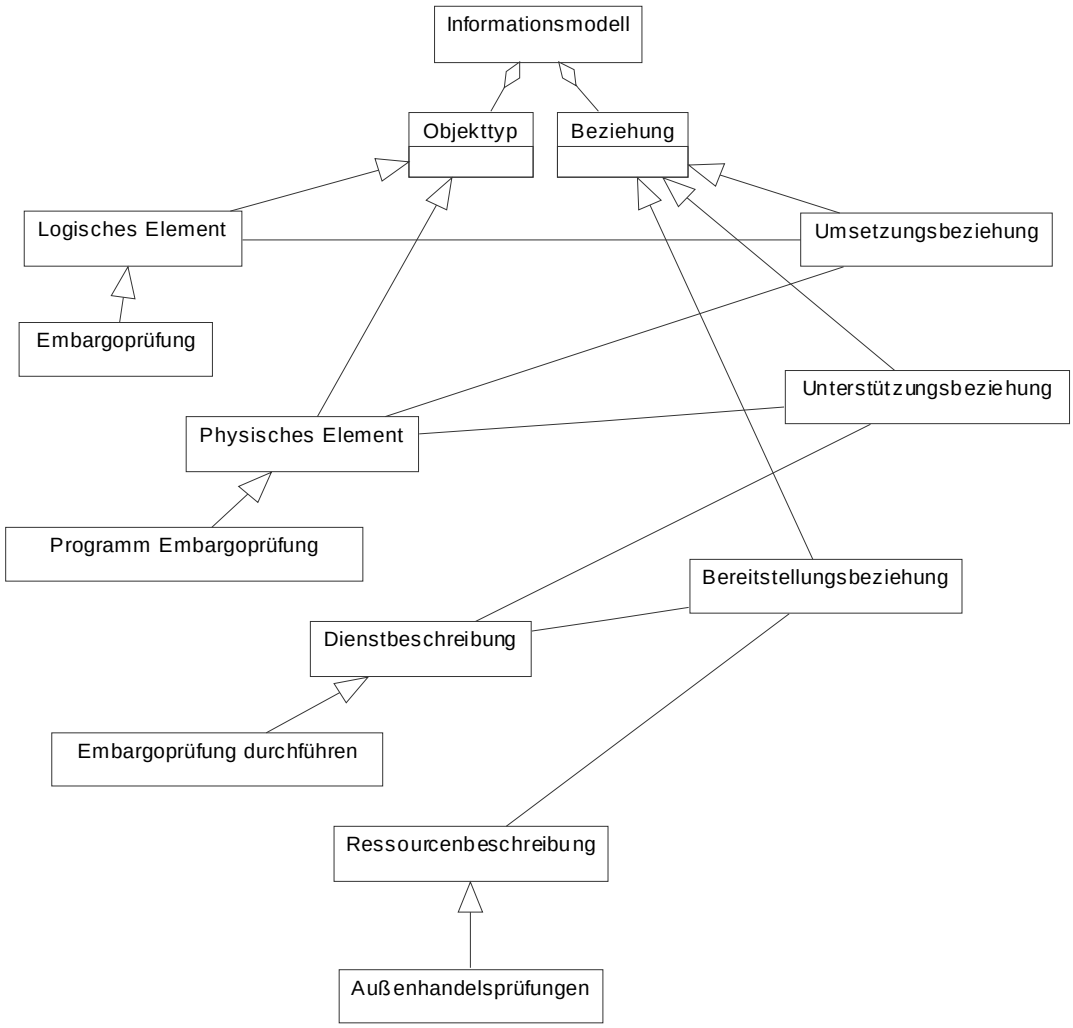


Abbildung 63: Erweiterungen im Wörterbuch

Die Durchführung der Erweiterung um ein logisches und ein physisches Element sowie die dazugehörigen Dienste und Ressourcen hat gezeigt, daß durch das entwickelte Workflow-Modell-Wörterbuch das angestrebte Ziel des Kapitels erreicht wurde. Nun kann untersucht werden, welche nicht-funktionalen Anforderungen sich aus dem Workflow-Modell-Wörterbuch ergeben.

7.4 Nicht-funktionale Anforderungen

Die Aufnahme von Dienst- und Ressourcenbeschreibungen in das Workflow-Modell-Wörterbuch schlägt sich in zwei nicht-funktionalen Anforderungen nieder. Dargestellt ist dies in Abbildung 64.

Die Möglichkeit, zusätzliche Dienstbeschreibungen aufzunehmen, resultiert in der Anforderung der Diensteerweiterbarkeit. Sie bedeutet, daß es möglich sein muß, zur Laufzeit des Software-Systems neue Dienste hinzuzufügen. Die Ressourcenerweiterbarkeit leitet sich aus der Fähigkeit des Workflow-Modell-Wörterbuchs her, zusätzliche Ressourcenbeschreibungen zur Laufzeit aufzunehmen. Sie fordert, daß es möglich sein muß, zur Laufzeit zusätzliche Ressourcen zum Software-System hinzuzufügen.

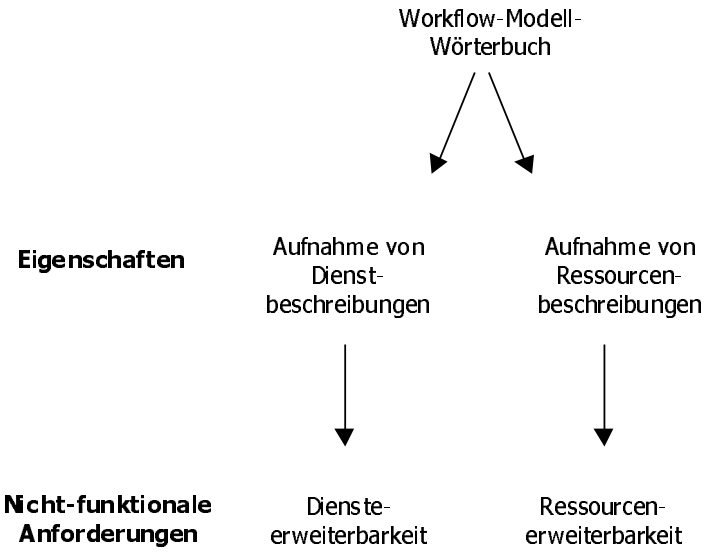


Abbildung 64: Anforderungen aus dem Workflow-Modell-Wörterbuch

7.5 Zusammenfassung

In diesem Kapitel wurde das Ziel der Schaffung einer dynamischen Verwaltung des Workflow-Modells durch ein Workflow-Modell-Wörterbuch erreicht. Dieses ermöglicht es, zur Laufzeit die Menge der von einem WfMS unterstützen Schemaelemente zu erweitern.

Hierzu wurden zunächst existierende Ansätze auf mögliche Lösungsbeiträge untersucht. Dabei stellte sich heraus, daß nur zwei Ansätze sich explizit mit dem Thema der Erweiterbarkeit beschäftigen und zudem das hier angestrebte Ziel verfehlen. Interessanter ist hingegen die Art und Weise, wie existierende Ansätze die Erweiterbarkeit um zusätzliche Schemata herstellen. Sie verwenden ein Wörterbuch zur Repräsentation von Workflow-Schemata.

Diese Idee wird aufgegriffen und die Schaffung eines Workflow-Modell-Wörterbuchs als Möglichkeit identifiziert, eine erweiterbare Repräsentation des Workflow-Modells zu schaffen. Der erste Schritt hierzu war die Entwicklung eines Informationsmodells. Als Grundlage wurde das im vorangegangenen Kapitel entwickelte Zwei-Ebenen-Workflow-Metamodell verwendet. In einem weiteren Schritt wurden Operationen zum Anlegen der Objekt- und Beziehungstypen definiert.

Aus dem Workflow-Modell-Wörterbuch konnten zwei nicht-funktionale Anforderungen abgeleitet werden. So schlägt sich die Aufnahme von Dienstbeschreibungen in der Anforderung der unabhängigen Diensteerweiterbarkeit nieder. Die Aufnahme von Ressourcenbeschreibungen wiederum resultiert in der Anforderung der unabhängigen Ressourcenerweiterbarkeit.

8 Aspektelementorientierte Schemarepräsentation

Ein wichtiges Ergebnis der Analyse in Abschnitt 5.1.1 war, daß sich die existierenden Ansätze in einem Dilemma zwischen Flexibilität und Skalierbarkeit befinden. So erreichen direkt Ausprägungen bildende Ansätze zwar eine hohe Flexibilität, bezahlen hierfür jedoch mit einer geringen Skalierbarkeit. Umgekehrt bieten die indirekt Ausprägungen bildenden Ansätze eine höhere Skalierbarkeit. Sie erreichen dies jedoch auf Kosten der Flexibilität.

Ein Ausweg aus diesem Dilemma kann durch einen indirekt Ausprägungen bildenden Ansatz gefunden werden, dessen Schemarepräsentation im Gegensatz zu den bisherigen Ansätzen dynamisch anpaßbar ist. Auf diese Weise wird gleichzeitig die Skalierbarkeit des indirekt Ausprägungen bildenden Ansatzes und die Flexibilität des direkt Ausprägungen bildenden Ansatzes erreicht.

Ziel dieses Kapitels ist es daher, die Schemarepräsentation so zu konzipieren, daß sie dynamisch an Schemaänderungen angepaßt werden kann. Es ist also zu klären, wie die Schemarepräsentation beschaffen sein muß, um die angestrebte Anpaßbarkeit zu erreichen. Dazu ist die Frage zu klären, wie die Schemarepräsentation auf einzelne Bestandteile aufzuteilen ist und in welchen Beziehungen diese Bestandteile stehen.

Ausgangspunkt ist die Erhebung, welche Arten von Schemaänderungen existieren, und die Analyse, in wieweit existierende Ansätze Beiträge liefern können. Da keiner der existierenden Ansätze eine Lösung liefert, wird dann schrittweise ein Lösungskonzept entwickelt. Ausgangspunkt ist die Zusammenführung von Aspektorientierter Programmierung (AOP) und den hier bereits dargestellten Aspekten von Workflow-Modellen, die in [ScAs98b] erstmalig beschrieben wurden. Diese führt zu einem ersten Lösungsansatz in Form der aspektorientierten Schemarepräsentation. Dieser kann jedoch noch nicht alle Anforderungen erfüllen, daher folgt eine Verfeinerung dieses Ansatzes zur sogenannten Aspektelementorientierten Schemarepräsentation, die die Anforderung der dynamischen Umsetzung von Workflow-Schemaänderungen voll erfüllt.

8.1 Existierende Ansätze

Um die potentiellen Beiträge existierender Ansätze identifizieren zu können, müssen zunächst die Anforderungen erhoben werden, die sich aus der Forderung nach einer dynamischen Umsetzung von Schemaänderungen ergeben. Dazu müssen die möglichen Schemaänderungen erfaßt und klassifiziert werden:

Die erste Gruppe von Schemaänderungen sind Veränderungen der Topologie der Schemaelemente. Sie werden kurz als *Topologieänderungen* bezeichnet. Es handelt sich also um Veränderungen der Anordnung der Schemaelemente. Es wird kein Schemaelement hinzugefügt oder gelöscht, die Menge der Schemaelemente bleibt gleich, lediglich ihre Anordnung wird verändert. Die zweite Gruppe von Schemaänderungen, *Parameteränderungen* genannt, ist die Veränderung der Parameter von Schemaelementen. Ein Beispiel ist die Veränderung des Grenzwerts, ab der eine detaillierte Auftragsprüfung durchgeführt wird. Die dritte Gruppe von Schemaänderungen besteht aus dem Hinzufügen von Schemaelementen, kurz *Schemaerweiterungen*. Die vierte Gruppe besteht aus dem Löschen von Schemaelementen. Derartige Änderungen werden als *Schemareduktionen* bezeichnet.

8.1.1 Monolithische Schemarepräsentation

Die einfachste Gestaltungsform einer Schemarepräsentation ist eine, bei der keine Zerlegung stattfindet: die monolithische Schemarepräsentation. Bei ihr ist weder eine Veränderung der Topologie der Schemaelemente, noch das Verändern einzelner Bestandteile noch das Löschen oder Hinzufügen von Schemaelementen ohne Neuerstellung der Schemarepräsentation möglich. Aus diesem Grund unterstützt die monolithische Schemarepräsentation keine der oben identifizierten Schemaänderungen.

8.1.2 Partitionierte Schemarepräsentation

Die einfachste Form der Zerlegung ist die Partitionierung der Schemarepräsentation. Sie findet sich beispielsweise im Meteor [DKMS96] Ansatz. Eine partitionierte Schemarepräsentation besteht aus mehreren Teilschemarepräsentationen, die jeweils ein Teilschema repräsentieren. Die Teilschemarepräsentationen können in einer zeitlichen Abfolge stehen, wie in Abbildung 65 dargestellt, müssen aber nicht.

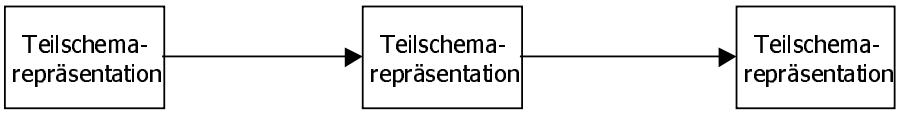


Abbildung 65: Partitionierte Schemarepräsentation

Durch die Partitionierung der Schemarepräsentation kann der Aufwand für die Umsetzung von Änderungen reduziert werden, da nur noch ein Teil der Schemarepräsentation neu erstellt werden muß. Änderungen und Erweiterungen des Workflow-Schemas bleiben jetzt auf eine Teilschemarepräsentation beschränkt. Es muß also nur die betroffene Teilschemarepräsentation ausgewechselt werden. Die Partitionierung erlaubt zwar eine Reduzierung des Änderungsaufwandes. Für jede Teilschemarepräsentation verbleibt jedoch ein hoher Änderungsaufwand, da jede Teilschemarepräsentation für sich betrachtet eine monolithische Schemarepräsentation darstellt und daher den gleichen Einschränkungen wie diese unterliegt.

8.1.3 Objektorientierte Schemarepräsentation

Eine objektorientierte Schemarepräsentation ist in mehreren Ansätzen wie beispielsweise WorCos [Schu99], [LeRo97], [Wesk98], [WHKS98] eingesetzt worden. Dabei werden die Workflows als Workflow-Klassen repräsentiert. Allerdings geschieht dies unter verschiedensten Namen wie beispielsweise als „process object“. Mitunter werden, wie im WorCos-Ansatz, spezielle Klassen für die Darstellung von Elementen des Operations- und Verzeichnisaspekts verwendet, ohne jedoch eine weitergehende Zerlegung durchzuführen. Daher lassen sich in

WorCos und vergleichbaren Ansätzen Topologieveränderungen sowie Schemaerweiterungen und -reduktionen nur für Aspekte des Operations- und Verzeichnisaspektes durchführen, nicht jedoch für andere Aspekte. Parameteränderungen sind bei keinem Aspekt möglich.

Dieses negative Ergebnis überrascht zunächst, ist die Objektorientierung doch unbestreitbar von allgemeinem Nutzen. Die tiefere Ursache soll daher durch eine detaillierte Untersuchung ermittelt werden.

Kernkonzept der Objektorientierung ist eine funktionale Zerlegung und die Abstraktion von Objekten in Form von Klassen. Klassen stellen dabei semantische Einheiten dar, die untereinander in Vererbungsbeziehungen stehen. Sie beschreiben die Methoden und Attribute von einem oder mehreren Objekten, die zum Ausführungszeitpunkt durch Exemplarbildung von ihnen gebildet werden, und kapseln deren Implementierung. Klassen stellen daher eine Laufzeitabstraktion dar.

Problematisch an der objektorientierten Zerlegung ist, daß bei ihr eine große Menge von quer zur Klassenstruktur verlaufender Funktionalität entsteht. So verlaufen Kontroll- und Datenflüsse quer zu den Teil-Workflows. Veranschaulicht ist dies in Abbildung 66. Darüber hinaus gibt es eine Vielzahl weiterer Funktionalitäten wie beispielsweise die Statusverfolgung, die Statusaufzeichnung, Fehlerbehandlung usw., die über die am Workflow beteiligten Klassen verteilt sind.

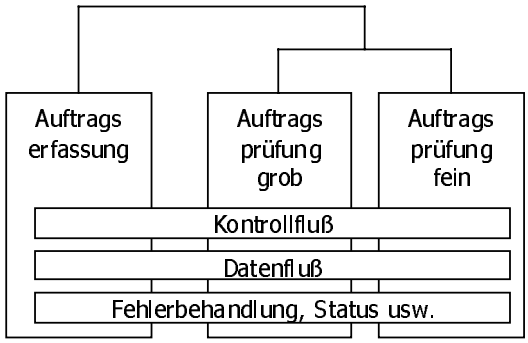


Abbildung 66: Objektübergreifende Funktionalität in Workflows

Änderungsoperationen an einem Workflow-repräsentierenden Objekt haben durch das Vorhandensein objektübergreifender Funktionalität in hohem Maße Seiteneffekte auf andere Objekte. Wird beispielsweise der Kontrollfluß in einer Klasse verändert, so müssen entsprechende Änderungen an allen verbundenen Klassen konsistent durchgeführt werden.

Diese bei Workflows und auch in anderen Bereichen auftauchenden Probleme haben dazu geführt, daß die Art und Weise der Zerlegung der Anwendungsfunktionalität mit Hilfe der Objektorientierung in letzter Zeit zunehmend hinterfragt wird [AkBe92], [AWBB93], [LoWa95], [Kicz96]. Dabei stellt sich heraus, daß die Objektorientierung wenig für Anwendungen geeignet ist, die ein hohes Maß an Funktionalität aufweisen, die quasi orthogonal zur Klassenstruktur auf mehrere Klassen verteilt ist. Ein Beispiel sind Interaktionsprotokolle zwischen Objekten [LoWa95]. Diese werden über mehrere Objekte verteilt, woraus sich eine Reihe von Problemen ergeben. So muß dafür gesorgt werden, daß in allen Objekten immer das gleiche Interaktionsprotokoll implementiert ist. Verallgemeinert man diese Betrachtung, so sieht man, daß außer Interaktionsprotokollen noch eine Vielzahl anderer Formen von Anwendungsfunktionalität existieren, die quasi „quer“ zur Klassenstruktur angeordnet sind. Beispiele sind Verteilung, Synchronisation und Fehlerbehandlung [Kicz96]. Dafür wurde der Begriff „cross-cutting functionality“ [KILL97] geprägt.

8.2 Konzeption

In Erkenntnis der oben beschriebenen Probleme sind eine Reihe von Ansätzen zu deren Beseitigung entstanden. Allianzen [LoWa95] kapseln Interaktionsprotokolle zwischen Objekten. Der „Composition Filters“-Ansatz [AkBV92] hat zum Ziel, Koordinationsvorgänge in objektorientierten Systemen besser zu unterstützen. Dazu wird die Verteilung von Code vermieden, der zur Unterstützung von Koordinationsvorgängen auf mehrere Objekte dient. Eine stärkere Trennung des Verhaltens von Objekten von deren Struktur strebt der Demeter-Ansatz [Lieb96] an. Die bisher genannten Ansätze beschränken sich jedoch auf eine Art von quer zur Objektstruktur verlaufender Funktionalität.

8.2.1 Aspektorientierte Schemarepräsentation

Die Aspektorientierte Programmierung (Aspect-Oriented-Programming) [Kicz96] verallgemeinert das Vorgehen von Ansätzen, wie beispielsweise den Allianzen oder der Composition Filters: Sie stellt ein allgemeines Konzept für die Unterstützung von quer zur Klassen- und Objektstruktur liegender Funktionalität bereit. Dazu wird die quer zur Klassen- und Objektstruktur liegende Funktionalität in Form von Aspekten erfaßt und separat von der Klassen- und Objektstruktur behandelt. In diesem Zusammenhang ist es wichtig, darauf hinzuweisen, daß die Aspekte der Aspektorientierten Programmierung zunächst unabhängig von den Aspekten in Workflow-Metamodellen entstanden sind. Die Zusammenführung der Begriffe erfolgt erstmalig in [ScAs98c].

Im Rahmen der Aspektorientierten Programmierung wird nicht mehr nur ein Aspekt, wie beispielsweise Interaktionsprotokolle, betrachtet, sondern es wird versucht, alle Funktionalitäten zu erfassen, die durch die Klassenstruktur nur unzureichend erfaßt werden. Das Konzept der Aspektorientierten Programmierung führt also die separate Behandlung für alle Aspekte durch. Die Aspektorientierte Programmierung ist nicht als Ersatz, sondern als Ergänzung zur Objektorientierung zu verstehen.

Überträgt man das Konzept der Aspektorientierten Programmierung auf die Gestaltung der Schemarepräsentation, wie dies erstmalig in [SABK98] geschah, so wird die Funktionalität der Schemarepräsentation in Abhängigkeit davon, welchen Aspekt sie unterstützt, getrennt. Es findet nicht nur eine funktionale Zerlegung statt wie bei der Objektorientierung oder eine Zerlegung in Klassen und einen separaten Aspekt wie bei Allianzen, sondern es wird eine Zerlegung in eine Vielzahl von Aspekten vorgenommen. Trotzdem ist eine derartige Zerlegung noch nicht fein genug, um die erwünschten Änderungsoperationen des Workflow-Schemas umsetzen zu können: Weder ist es möglich, die Topologie des repräsentierten Workflow-Schemas zu ändern, noch können Schemaelemente ergänzt bzw. entfernt werden. Der Grund hierfür liegt in der Zusammenfassung aller Elemente eines Aspekts in einer Aspektrepräsentation.

8.2.2 Aspektelementorientierte Schemarepräsentation

Die Einschränkungen der aspektorientierten Zerlegung beseitigt die aspektelementorientierte Zerlegung, die in [ScAs98a], [ScAs98b], [ScAs98c] eingeführt wurde. Sie führt eine Zerlegung der Schemarepräsentation in Bestandteile durch, die jeweils nur ein Aspektelement repräsentieren. Diese Bestandteile werden als *Repräsentationselemente* bezeichnet. Ein Repräsentationselement ist also ein Bestandteil einer Schemarepräsentation, das genau ein Schemaelement repräsentiert. Die Repräsentationselemente sind über sogenannte *Verbinder* miteinander verbunden. Es werden zwei Arten von Verbindern unterscheiden. Die temporalen Verbinder geben zeitliche Abfolgen zwischen den Repräsentationselementen wieder. Nicht-temporale Verbinder geben Dienstnehmer-/Dienstgeberbeziehungen wieder. Die Verbindung zweier Repräsentationselemente über einen Verbinder setzt voraus, daß beide über entsprechende *Verbindungspunkte*

verfügen. Entsprechend der Einteilung bei den Verbindern wird zwischen temporalen und nicht-temporalen Verbindungspunkten unterschieden.

Die Darstellung der Repräsentationselemente erfolgt durch Rechtecke, wie in Abbildung 67 an einem Beispiel dargestellt. Die Repräsentationselemente A und B verfügen über temporale Verbindungspunkte und sind über einen temporalen Verbinder verbunden. Die Repräsentationselemente A und C verfügen über nicht-temporale Verbindungspunkte und sind über einen nicht-temporalen Verbinder verbunden.

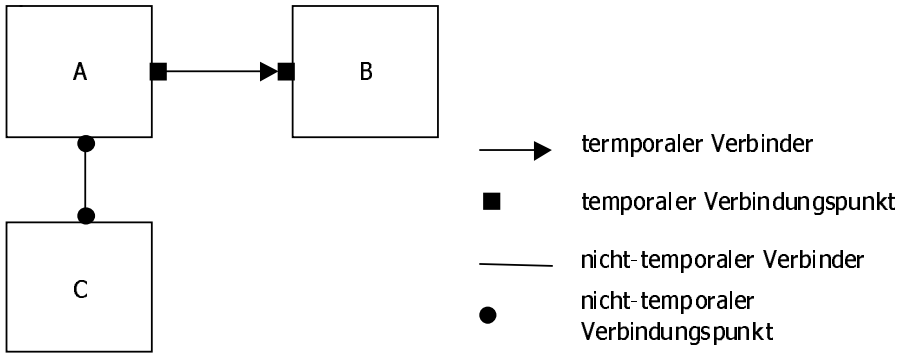


Abbildung 67: Repräsentationselemente, Verbinder und Verbindungspunkte

Zur Veranschaulichung der aspektelementorientierten Zerlegung dient das bereits aus früheren Beispielen bekannte Workflow-Schema. Die auf seiner Basis entwickelte Schemarepräsentation ist in Abbildung 68 dargestellt. Entsprechend der Konzeption der aspektelementorientierten Zerlegung wird für jedes Element des Workflow-Schemas ein Repräsentationselement geschaffen. Es handelt sich um die Repräsentationselemente "Programm Auftragserfassung", "XOR-Split", "Auftragsspeicher", "Programm Standardprüfung", "Programm Detaillierte Prüfung" und "OR-Join" sowie „Programm Weiterverarbeitung“. Die Repräsentationselemente entsprechen von ihrer Granularität den Elementen des Workflow-Schemas. Die Repräsentationselemente „Programm Auftragserfassung", "Programm Standardprüfung", "Programm Detaillierte Prüfung" und „Programm Weiterverarbeitung“ repräsentieren Schemaelemente, die dem Operationsaspekt zugehörig sind. Die Repräsentationselemente "XOR-Split" und "OR-Join" repräsentieren Schemaelemente, die dem Kontrollaspekt zugehörig sind. Das Repräsentationselement "XOR-Split" ist parametrisierbar, indem der Name des Entscheidungsparameters und der Entscheidungswert angegeben werden kann. Hier ist also der Name des Entscheidungsparameters „Auftragswert“ und der Entscheidungswert 5000. Das Repräsentationselement "Auftragsspeicher" ist schließlich dem Datenaspekt zugehörig.

Die Repräsentationselemente sind über temporale und nicht-temporale Verbinder verbunden. Zur Vereinfachung der Bezugnahme im Text sind die temporalen Verbinder mit den Buchstaben a bis f und die nicht-temporalen Verbinder mit den Ziffern 1 bis 4 identifiziert. Insgesamt besteht die aspektelementorientierte Schemarepräsentation aus sieben Repräsentationselementen, 4 nicht-temporalen und 6 temporalen Verbindern.

Der temporale Verbinder a besteht zwischen dem Repräsentationselement "Programm Auftragserfassung" und "XOR-Split". Dadurch wird zum Ausdruck gebracht, daß nach Ausführung "Programm Auftragserfassung" das Repräsentationselement "XOR-Split" zur Ausführung kommen soll. Der nicht-temporale Verbinder 1 besteht hingegen zwischen den Repräsentationselementen "Programm Auftragserfassung" und "Auftragsspeicher". Hierdurch wird wiedergegeben, daß das Repräsentationselement "Programm Auftragserfassung" das Repräsentationselement "Auftragsspeicher" im Rahmen einer Dienstnehmer-

/Dienstgeberbeziehung nutzt. Dies kann in einem Schritt erfolgen, kann aber auch mehrere Schritte erforderlich machen.

Das Repräsentationselement "XOR-Split" ist über die temporalen Verbinder b und c mit den Repräsentationselementen "Programm Standardprüfung" bzw. "Programm Detaillierte Prüfung" verbunden. Das Repräsentationselement "XOR-Split" prüft den Wert des Auftrags und aktiviert in Abhängigkeit davon, ob der Auftrag einen Wert kleiner 5000 DM hat oder nicht, entweder das Repräsentationselement "Programm Standardprüfung" oder "Programm Detailprüfung". Beide Repräsentationselemente sind über die nicht-temporalen Verbinder 3 und 4 mit dem Auftragsspeicher verbunden, um die Prüfungen durchzuführen. Schließlich sind sie über die temporalen Verbinder d und e mit dem Repräsentationselement "OR-Join" verbunden. Das Repräsentationselement „OR-Join“ ist über den temporalen Verbinder f mit dem Repräsentationselement „Programm Weiterverarbeitung“ verbunden.

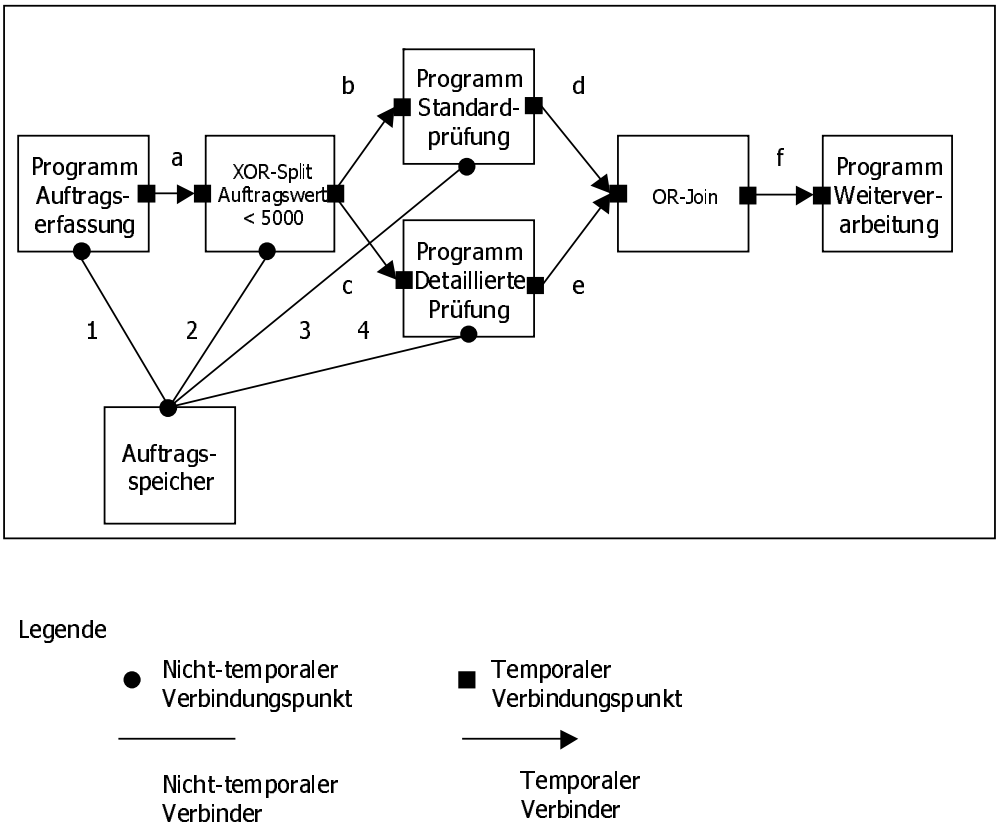


Abbildung 68: Aspektelementorientierte Zerlegung

8.3 Bewertung

Nun wird untersucht, ob die aspektelementorientierte Schemarepräsentation die Durchführung von Schemaänderungen unterstützt. Es gilt also die Frage zu klären, ob die aspektelementorientierte Schemarepräsentation in der Lage ist, die eingangs identifizierten Formen von Schemaänderungen, also Topologie- und Parameteränderungen sowie Schemaerweiterungen und –reduktionen zu unterstützen. Zunächst wird dies auf einer abstrakten Ebene untersucht, um dann an Hand eines anschaulichen Beispieles die Durchführbarkeit dieser Formen der Schemaänderungen zu demonstrieren.

Die erste Gruppe von Schemaänderungen, die untersucht werden, sind Topologieänderungen. Sie können in einer aspektelementorientierten Schemarepräsentation leicht durch Entfernen und Hinzufügen von Verbindern umgesetzt werden, ohne daß die Repräsentationselemente hiervon

beeinflußt würden. Die zweite Gruppe von Schemaänderungen, die der Parameteränderungen, kann durch die Anpassung von Repräsentationselementen umgesetzt werden. Schemaerweiterungen und -reduktionen sind ebenfalls leicht durchzuführen, da die Granularität der Repräsentationselemente denen des Schemas entspricht.

Zur Veranschaulichung wird eine Erweiterung des Beispielschemas umgesetzt. Die Erweiterung besteht im Hinzufügen einer Embargoprüfung: Nach der durchgeführten Detail- bzw. Standardprüfung findet ein Test statt, ob der Auftraggeber aus einem Embargoland stammt. In Abhängigkeit vom Ergebnis wird entweder die Bearbeitung fortgesetzt werden oder eine Absage verschickt.

Zur Umsetzung dieser Schemaänderung werden zunächst die Repräsentationselemente „Embargoprüfung“ und „Programm Absage versenden“ neu eingeführt. Es findet also eine Schemaerweiterung statt. Beide sind dem Operationsaspekt zugehörig. Zur Einbindung der beiden Repräsentationselemente ist außerdem eine Topologieveränderung notwendig. Kein neues Repräsentationselement ist hingegen für die Durchführung des XOR-Splits notwendig. Das bereits vorhandene Repräsentationselement kann für die zweite Verwendung passend parametrisiert werden. Dazu wird als Entscheidungsparameter „Embargo“ und der Entscheidungsparameter „ok“ angegeben.

Zur Umsetzung der Schemaänderung wird das Repräsentationselement „OR-Join“ über den temporalen Verbinder g mit dem Repräsentationselement „Embargoprüfung“ verbunden. Das Repräsentationselement „Embargoprüfung“ ist außerdem über den nicht-temporalen Verbinder 6 mit dem Repräsentationselement „Auftragsspeicher“ verbunden. Zur Weiterverarbeitung ist das Repräsentationselement „Embargoprüfung“ über den temporalen Verbinder h mit dem Repräsentationselement „XOR-Split“ verbunden. Dieses wiederum ist über die temporalen Verbinder i und j mit den Repräsentationselementen „Programm Weiterverarbeitung“ und „Programm Absage versenden“ verbunden. Über den nicht-temporalen Verbinder 7 ist das Repräsentationselement XOR-Split außerdem mit dem Repräsentationselement „Auftragsspeicher“ verbunden. Die dementsprechend veränderte aspektelementorientierte Schemarepräsentation findet sich in Abbildung 69.

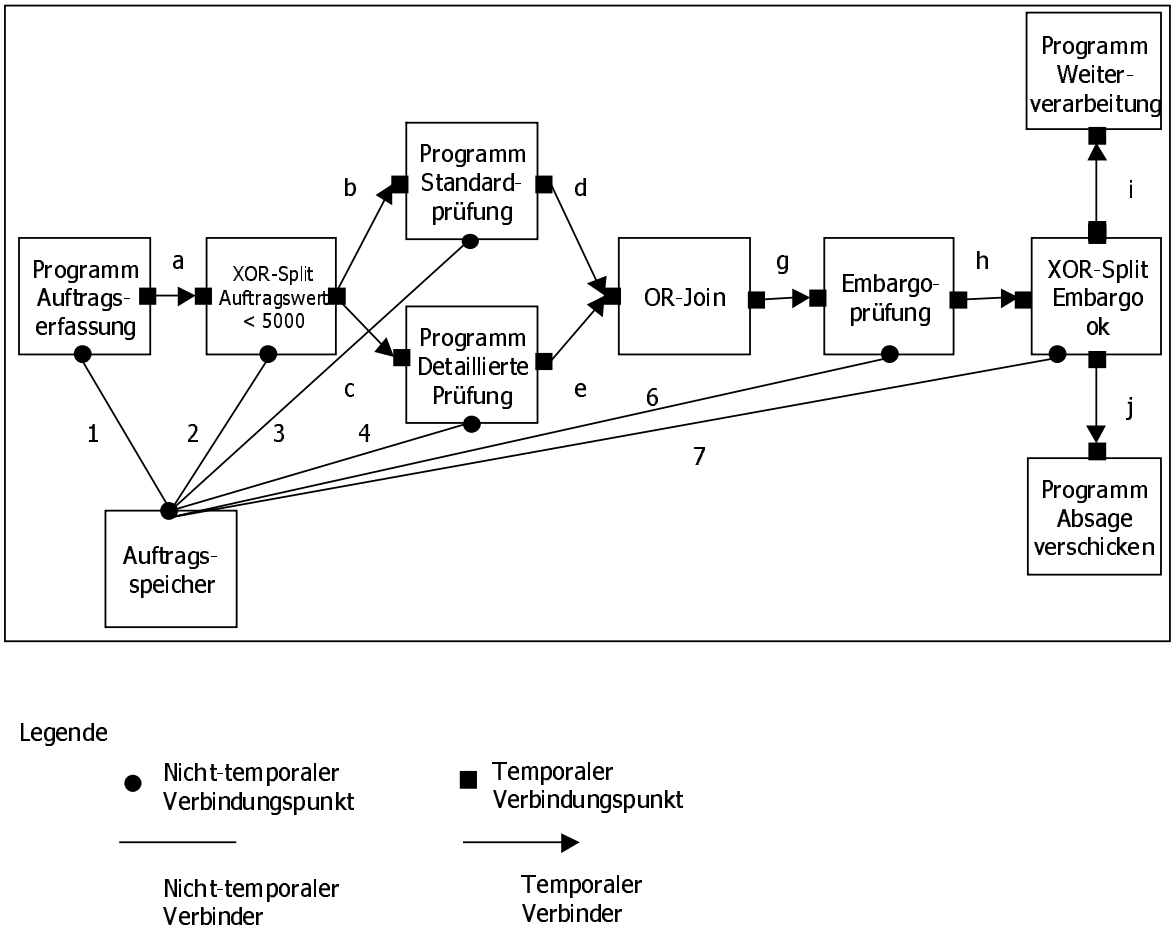


Abbildung 69: Erweiterte Schemarepräsentation

Die aspektelementorientierte Zerlegung der Schemarepräsentation vermeidet die Nachteile der aspektorientierten Zerlegung. Durch das feinere Zerlegungsgranulat werden von Änderungen nur die Repräsentationselemente betroffen, die von der Änderung betroffene Schemaelemente repräsentieren. Auf diese Weise können alle Formen von Änderungen also Topologie- und Parameteränderungen sowie Schemaerweiterungen und –reduktionen leicht ausgeführt werden. Als Fazit ist daher zu ziehen, daß die aspektelementorientierte Schemarepräsentation die an sie gestellten Anforderungen erfüllt.

8.4 Nicht-funktionale Anforderungen

Nun wird untersucht, welche nicht-funktionalen Anforderungen sich aus der aspektelementorientierten Schemarepräsentation ergeben. Betrachtet wird das Hinzufügen und Entfernen von Verbindern, das Hinzufügen und Entfernen von Repräsentationselementen sowie die Anpassung von Repräsentationselementen durch Parametrisierung. Die Ableitung der Anforderungen ist in Abbildung 70 dargestellt.

Damit das Hinzufügen und Entfernen von Verbindern der Schemarepräsentation umgesetzt werden kann, müssen auch Verknüpfungen von Diensten zur Laufzeit erstell- und zerstörbar sein. Es muß also möglich sein, zur Laufzeit, d.h. ohne erneute Erstellung der Ressourcen, die Dienste so zu verändern, daß Verknüpfungen neu erstellt bzw. zerstört werden können. Diese Anforderung wird als *Anwendungskonfigurierbarkeit* bezeichnet. Ein Beispiel hierfür ist die Löschung des temporalen Verbinders h und des nicht-temporalen Verbinders 5 sowie die Neuerstellung der temporalen Verbinder g bis j und der nicht-temporalen Verbinder 6 und 7.

Um das Hinzufügen und Entfernen von Repräsentationselementen zu unterstützen, müssen auch Dienste entfernt bzw. hinzugefügt werden können. Diese Anforderung wird als *Anwendungserweiterbarkeit* bezeichnet. Ein Beispiel dafür ist die Hinzunahme der Ressourcen für die Embargoprüfung und das Versenden einer Absage.

Um die Veränderung von einzelnen Repräsentationselementen umsetzen zu können, muß das Software-System in der Lage sein, zur Laufzeit die von Ressourcen bereitgestellten Dienste anpassen zu können. Dies schlägt sich in der Anforderung der *Dienstekonfigurierbarkeit* nieder. Sie bezeichnet die Anforderung, daß die Eigenschaften der Dienste zur Laufzeit angepaßt werden können. Das Beispiel hierfür ist die Anpassung des Repräsentationselementes XOR-Split durch Angabe eines Entscheidungsparameters und eines Entscheidungswerts.

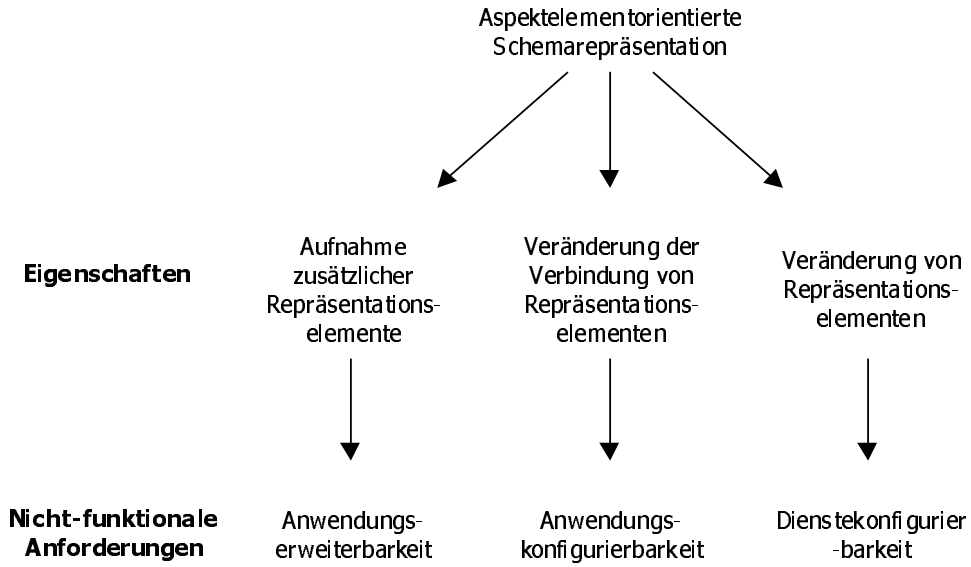


Abbildung 70: Nicht-funktionale Anforderungen

8.5 Zusammenfassung

Ziel dieses Kapitels war es, die Schemarepräsentation so zu konzipieren, daß sie dynamisch an Schemaänderungen angepaßt werden kann. Dieses Ziel wurde mit der aspektelementorientierten Schemarepräsentation erreicht.

Zunächst wurden existierende Ansätze auf mögliche Beiträge untersucht. Als Kriterium dienten dabei vier Gruppen von Änderungen auf Workflow-Schemata: Topologie- und Parameteränderungen sowie Schemaerweiterungen- und –reduktionen. Es stellte sich jedoch heraus, daß keiner dieser Ansätze eine Lösung der gestellten Problematik erbrachte. Daher wurde über den Zwischenschritt der aspektorientierten Schemarepräsentation die aspektelementorientierte Schemarepräsentation konzipiert. Die aspektelementorientierte Schemarepräsentation zerlegt die Schemarepräsentation in Repräsentationselemente die über Verbindungspunkte und Verbinder verknüpft sind. Jedes der Repräsentationselemente gibt jeweils nur ein Aspektelement einer Schemarepräsentation wieder. Die Verbinder geben entweder temporale oder nicht-temporale Abhängigkeiten zwischen den Repräsentationselementen wieder. Die durchgeführte Bewertung zeigte, daß die aspektelementorientierte Schemarepräsentation die an sie gestellten Anforderungen erfüllt.

Aus der aspektelementorientierten Schemarepräsentation wurden drei nicht-funktionale Anforderungen abgeleitet. Die Aufnahme zusätzlicher Repräsentationselemente resultierte in der An-

forderung der Anwendungserweiterbarkeit. Die Veränderung der Verbindung von Repräsentationselementen führte zur Anforderung der Anwendungskonfigurierbarkeit. Schließlich wurde aus der Veränderung von Repräsentationselementen die Anforderung der individuellen Dienstekonfigurierbarkeit abgeleitet.

9 Komponentensysteme als Realisierungsgrundlage

In den vorangegangenen Kapiteln wurden drei Lösungskonzepte für die informationstechnische Unterstützung weitreichender Geschäftsprozesse geschaffen. Aus diesen Lösungskonzepten wurden sechs nicht-funktionale Anforderungen abgeleitet, die von einer Software-Architektur erfüllt werden müssen, damit sie zur Realisierung der Lösungskonzepte geeignet ist. Diese Anforderungen sind:

- Anwendungs-, Dienste- und Ressourcenkonfigurierbarkeit
- Anwendungs-, Dienste- und Ressourcenerweiterbarkeit

Ziel dieses Kapitels ist es, mit Komponentensystemen eine Architektur vorzustellen, die alle diese Anforderungen erfüllt. Komponentensysteme setzen sich aus drei Bestandteilen zusammen: Komponenten, komponentenorientierten Rahmenwerken und kompositen Anwendungen.

Bisher existieren bereits die Konzepte der Komponente und des komponentenorientierten Rahmenwerks. Bei ihrer Darstellung besteht die Herausforderung darin, daß kein begriffliches Instrumentarium bereitsteht, welches erlaubt zu beurteilen, ob und in wieweit Komponenten und komponentenorientierte Rahmenwerke die nicht-funktionalen Anforderungen erfüllen. Zwar gibt es eine Vielzahl von Veröffentlichungen, die sich mit Komponenten beschäftigen. Diese haben jedoch — so wird sich zeigen — stark differierende Vorstellungen vom Begriff der Komponente [Adle95], [Jaza95], [Deri96], [Meye99], [Ober96c], [FSHS97], [NiLu97], [MeMi99], [Kraj]. Zusätzlich zeichnet sich ab, daß eine zweite Generation von Komponentensystemen im Entstehen ist [Assm97], [AsSc97].

Um diesem Mißstand abzuhelfen, wird daher ein Begriffssystem geschaffen. Mit ihm kann geklärt werden, welche der nicht-funktionalen Anforderungen von Komponenten und komponentenorientierten Rahmenwerken erfüllt werden. Es sind dies die Anforderungen der Dienste- und Ressourcenkonfigurierbarkeit sowie der Dienste- und Ressourcenerweiterbarkeit. Komponenten und komponentenorientierte Rahmenwerke erfüllen jedoch nicht die Anforderung der Anwendungskonfigurier- und –erweiterbarkeit.

Aus diesem Grund werden komposite Anwendungen als neues Anwendungsparadigma für die Verwendung von Komponenten und komponentenorientierten Rahmenwerken vorgestellt. Komposite Anwendungen wurden erstmalig in [Schm97a] beschrieben und werden hier in einer erweiterten Form dargestellt. Sie erfüllen die Anforderungen der unabhängigen Anwendungserweiterbarkeit und –konfigurierbarkeit. Daher können Komponentensysteme als Architektur

aus Komponenten, komponentenorientierten Rahmenwerken und kompositen Anwendungen alle aufgestellten Anforderungen erfüllen.

9.1 Komponenten

Über die Zeit hat sich eine Vielzahl von Ansätzen mit unterschiedlichen Auffassungen über Natur und Bestandteile einer Komponente herausgebildet. Die früheste Erwähnung findet sich in [McIl69]. Häufig wird jedoch der Begriff der Komponente auf Konzepte angewandt, in denen eine Komponente lediglich eine syntaktische Klammer für einen unbestimmten Teil eines Gesamtsystems darstellt wie beispielsweise in [NiTs95]. Eine weitere Verwendung des Begriffs Komponente ist die im Rahmen von Architekturdefinitionssprachen [GaSh93], [GaSh94], [NiDa95], [Medv97a], [Medv97b] oder Architekturstilen [MKMG97]. Komponenten sind hier abstrakte architekturelle Einheiten. Eine ausführliche Diskussion der Unterschiede dieses Komponentenbegriffs zu dem in Technologien wie ActiveX/DCOM [Broc96], [Chap96], [Box98], [COM], CORBA [OMG] oder Enterprise Java Beans [EJB] findet sich in [OMTR98]. Schließlich wird unter dem Begriff einer Komponente auch eine Marketingeinheit für Software verstanden. Darüber hinaus gibt es eine Vielzahl weiterer Arbeiten, die ebenfalls den Begriff der Komponente erwähnen, allerdings ohne eine klare Definition zu geben.

Die Definition, die in dieser Arbeit entwickelt werden wird, leitet sich vom Ziel ab, Software-Systeme um unabhängig entwickelte Dienste und Ressourcen erweitern zu können und die unabhängige Weiterentwicklung dieser Dienste und Ressourcen zu unterstützen.

Damit unabhängig entwickelte Dienste und Ressourcen genutzt werden können, müssen die Bedingungen für die Nutzung dieser unabhängigen Dienste und Ressourcen bekannt sein. Diese Bedingungen werden als Kontextabhängigkeiten bezeichnet [CiSc96]. Sie können beispielsweise im Vorhandensein einer bestimmten Ablaufumgebung oder einer bestimmten Bibliothek bestehen. Kontextabhängigkeiten können explizit oder implizit sein. Explizite Kontextabhängigkeiten können in Form einer benötigten Schnittstelle angegeben werden. Implizite Kontextabhängigkeiten liegen beispielsweise vor, wenn eine Bibliothek nur unter Kenntnis ihres Quellcodes genutzt werden kann.

Will man unabhängig entwickelte Dienste und Ressourcen nutzen, so sollten diese nur explizite Kontextabhängigkeiten aufweisen. Prinzipiell ist auch die Nutzung von Diensten oder Ressourcen denkbar, die implizite Kontextabhängigkeiten aufweisen, der Aufwand zur Ermittlung und Überwindung dieser Kontextabhängigkeiten übersteigt dann aber häufig den erzielten Gewinn durch die Wiederverwendung.

Ein Mechanismus, der in hohem Maße implizite Kontextabhängigkeiten erzeugt, ist die Implementierungsvererbung [Szyp92], [Taiv96], [Szyp97], [Szyp98], [Box98], [Weck96]. Bei der Implementierungsvererbung wird die Implementierung der Methoden der vererbenden Klasse an erbende Klassen weitergegeben. Dabei wird jedoch die Kapselung der Objekte durchbrochen [PfSz96], [WeSz96]. Es entstehen implizite Kontextabhängigkeiten, was ein Grund dafür ist, daß der von der Objektorientierung versprochene „Objektmarkt“ mit in Katalogen angebotenen Objekten [Cox90] sich nicht verwirklicht hat [Nier91], [Udel94]. Unter dem Begriff des „fragile base class problem“ [PfSz96] ist diese Problematik Gegenstand der Forschung [MiSe98]. Daher verzichten Komponenten auf die Implementierungsvererbung und verwenden alleine die Schnittstellenvererbung. Die Schnittstellenvererbung führt nur die Definition polymorpher Unterklassen durch, die Vererbung der Implementierungen findet nicht statt.

Der zweite, wichtige Mechanismus von Komponenten zur Vermeidung impliziter Kontextabhängigkeiten ist die strikte Kapselung. Sie bedeutet, daß alle Interaktionen der Kompo-

nente mit ihrer Umwelt über Schnittstellen abgewickelt werden. Dies geschieht, damit Komponenten zusammenarbeiten können, auch wenn sie aus verschiedenen Entwicklungskontexten stammen. Hierzu müssen jedoch die Abhängigkeiten vom jeweiligen Entwicklungskontext explizit gemacht werden. Auch muß es eine Übereinkunft über eine minimal zur Verfügung zu stellende Funktionalität geben. Dies bedeutet, daß eine Art von Vertrag zwischen dem Hersteller und dem Nutzer einer Komponente geschlossen werden muß, deren Gegenstand die Gestaltung der Schnittstellen der Komponente sind [BJPW99]. Als Ergebnis dieser Überlegungen ergibt sich folgende Definition einer Komponente[6]:

Definition 21 **Komponente**
Eine Komponente ist eine Software-Einheit, deren Schnittstellen Vertragscharakter haben und nur explizite Kontextabhängigkeiten aufweist [WCOP96].

Durch die Beschränkung auf explizite Kontextabhängigkeiten wird erreicht, daß eine Komponente leicht von einem Entwicklungskontext in eine Vielzahl von Anwendungskontexten übertragen werden kann. Auf diese Weise wird die Voraussetzung dafür geschaffen, Software-Systeme um unabhängig entwickelte Dienste und Ressourcen in Form von Komponenten zu erweitern.

Um implizite Kontextabhängigkeiten zu vermeiden, setzen Komponenten drei Mechanismen ein: Schnittstellen, Introspektions- und Spezialisierungsmechanismen. Sie werden im folgenden vorgestellt.

9.1.1 Schnittstellen

Komponenten können mehrere Schnittstellen besitzen. Dabei kann es sich um eingehende oder ausgehende Schnittstellen handeln. Eingehende Schnittstellen dienen zur Nutzung von Diensten, die von der Komponente angeboten werden. Ausgehende Schnittstellen erlauben der Komponente, externe Dienste zu nutzen. Eine Komponente kann eine Schnittstelle nur einmal besitzen, dies aber gleichzeitig in ein- und ausgehender Form. Dies bedeutet, daß eine Komponente beispielsweise einen Dienst einer anderen Komponente nutzt, ihn mit einem Mehrwert versieht und wiederum anderen Komponenten zur Nutzung anbietet. Dargestellt werden Komponenten und Schnittstellen mit Hilfe des in Abbildung 71 dargestellten Symbols. Eingehende Schnittstellen sind durch einen Kreis, ausgehende Schnittstellen durch eine Kreisfläche dargestellt.

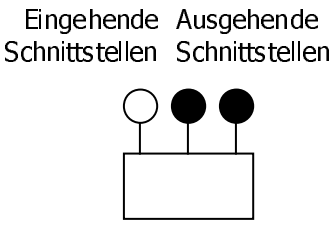


Abbildung 71: Komponentensymbol mit eingehenden und ausgehenden Schnittstellen

[6] An dieser Definition war der Autor beteiligt [CiSc96].

9.1.2 Ausprägungen

Eine Komponente kann wie eine Klasse Ausprägungen[7] bilden. Eine Komponentenausprägung ist eine eigenständige ausführbare Repräsentation der Komponente. Die Bildung von Ausprägungen ist nicht an die Verwendung einer Programmiersprache gebunden, sondern erfolgt sprachübergreifend. Während die Bildung von Ausprägungen bei einer C++ Klasse nur mit Hilfe C++-spezifischer Operatoren erfolgen kann, können Komponenten über Sprachgrenzen hinweg Ausprägungen bilden. Dazu werden die sprachspezifischen Mittel zur Bildung von Ausprägungen gekapselt.

Definition 22 **Komponentenausprägung**
Eine Komponentenausprägung ist eine eigenständig ausführbare Repräsentation der Komponente, die sprachunabhängig erzeugbar und sprachübergreifend nutzbar ist.

Komponentenausprägungen können zustandsbehaftet oder zustandslos sein. Ebenso können durch eine Komponentenausprägung persistente Informationen gekapselt sein. Komponentenausprägungen werden durch ein Komponentensymbol mit einer grauen Ausfüllung symbolisch dargestellt und durch ein Hochkomma gekennzeichnet, wie in Abbildung 72 veranschaulicht. $k1_1$‘ ist daher eine Ausprägung der Komponente $k1_1$.

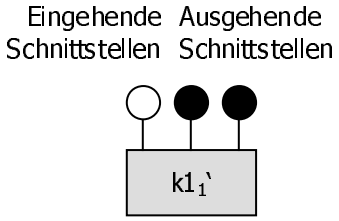


Abbildung 72: Symbol für Komponentenausprägungen

9.1.3 Introspektions- und Spezialisierungsmechanismen

Zwei weitere Mittel zur Vermeidung impliziter Kontextabhängigkeiten sind Introspektions- und Spezialisierungsmechanismen. Um Komponenten möglichst vielfältig einsetzbar zu machen, versucht der Komponentenentwickler so viele Implementierungsdetails wie möglich vor dem Komponentenanwender zu verbergen. Insbesondere werden Komponenten in bereits übersetzter Form ausgeliefert, so daß ihr Quellcode dem Komponentenanwender nicht bekannt ist. Für Komponenten müssen daher Anpassungstechniken verwendet werden, die ohne Quellcode auskommen. Bei Komponenten werden Introspektions- und Spezialisierungsmechanismen eingesetzt. Beide gehören zum Bereich der Metaprogrammierung [Assm97], auf die aber hier nicht weiter eingegangen wird.

Introspektionsmechanismen geben Auskunft über die von der Komponente bereitgestellte Funktionalität und anpaßbare Parameter. Erreichbar sind sie über eine Introspektionsschnittstelle, die im folgenden mit s_i gekennzeichnet wird. In Komponenten von COM (Component Object Model [COM]), EJB (Enterprise JavaBeans [EJB]) und dem CORBA Component Model [OMG] ist die Ausdrucksfähigkeit jedoch auf die Wiedergabe der syntaktischen Beschreibung der Komponente beschränkt. Eine komplexe semantische Beschreibung ist hingegen nicht möglich.

Spezialisierungsmechanismen erlauben eine Anpassung der Komponente, Spezialisierung genannt, ohne daß sie erneut übersetzt werden muß oder die Kenntnis ihrer Implementierung er-

[7] Ausprägung ist der Ersatz für das englische „instance“

forderlich ist. Die dabei verwendeten Informationen zur Anpassung der Komponente werden als Spezialisierungsinformationen bezeichnet. Die Spezialisierungsmechanismen sind über eine ausgezeichnete Schnittstelle, die Spezialisierungsschnittstelle, zugänglich. Für die Spezialisierungsschnittstelle wird im folgenden die Kennzeichnung s_{sp} verwendet werden. Über sie können Parameter oder das Verhalten der Komponente geändert werden. Spezialisierungsinformationen können transient oder persistent sein.

9.1.3.1 Generelle Spezialisierung

Die generelle Spezialisierung ist die einfachste Form der Spezialisierung. Bei ihr sind einer Komponente Spezialisierungsinformationen fest zugeordnet. Die Spezialisierung geschieht, indem eine Ausprägung der Komponente gebildet wird und unter Nutzung der Spezialisierungsschnittstelle s_{sp} die Spezialisierungsinformationen festgelegt und persistent gespeichert werden. Dargestellt ist dies in Abbildung 73.

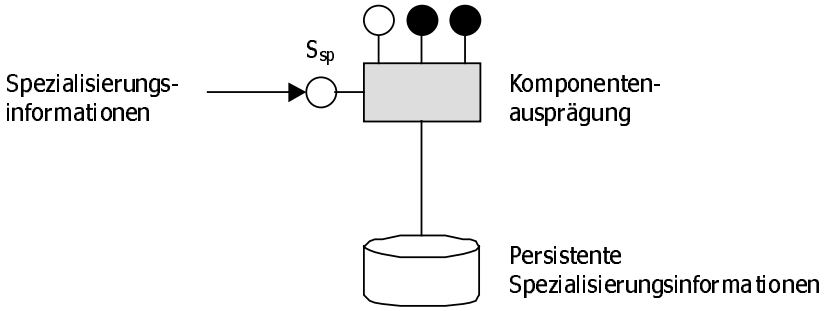


Abbildung 73: Generelle Spezialisierung

Auf der Basis dieser festgelegten Spezialisierungsinformationen werden alle Ausprägungen der Komponente gebildet. In Abbildung 74 wird gezeigt, wie die Ausprägungen der Komponente a, als a' und a" gekennzeichnet mit Hilfe der generellen Spezialisierung gebildet werden. Dazu lesen die Ausprägungen bei ihrer Entstehung die persistenten Spezialisierungsinformationen ein. Sie sind daher sofort nutzbar.

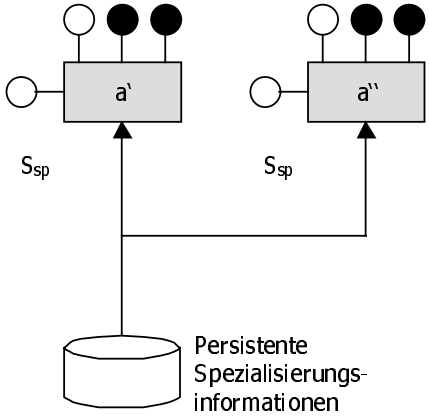


Abbildung 74: Bildung von Ausprägungen bei genereller Spezialisierung

Der große Nachteil der generellen Spezialisierung ist, daß eine Komponente nur in einer Anwendung verwendbar ist. Möchte man die Komponente in unterschiedlichen Anwendungen einzusetzen, so ist es notwendig, Kopien von ihr anzufertigen und diese mit separaten Spezialisierungsinformationen auszustatten. Die Verwendung derselben Komponente in unterschiedlichen Anwendungen ist nicht möglich.

9.1.3.2 Individuelle Spezialisierung

Die individuelle Spezialisierung vermeidet diese Nachteile. Bei ihr wird die Spezialisierung nicht einmalig durchgeführt, sondern die Spezialisierung wird für jede Komponentenausprägung separat durchgeführt. Die Spezialisierungsinformationen werden nicht mehr bei der Komponente gespeichert, sondern von außen bereitgestellt. Dies kann beispielsweise durch eine externe Anwendung geschehen.

In Abbildung 75 ist zu sehen, wie zwei Ausprägungen der Komponente a, bezeichnet als a' und a", durch Anwendungen mit unterschiedlichen Spezialisierungsinformationen versehen werden. Dies wird an Hand eines Parameters Auftragswert veranschaulicht, dessen Wert durch Spezialisierung festgelegt wird. Durch Spezialisierung kann für die Komponentenausprägung a' der Wert 5000 und für die Komponentenausprägung a'' der Wert 6000 angegeben werden. Auch hier wird die Spezialisierungsschnittstelle s_{sp} genutzt.

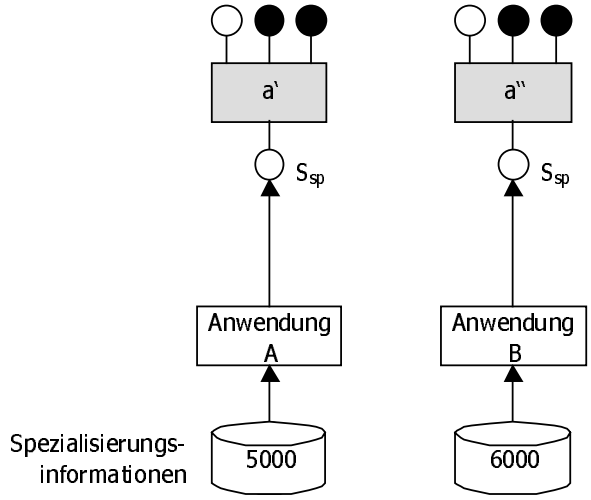


Abbildung 75: Individuelle Spezialisierung

Die Vorteile der individuellen Spezialisierung liegen in der Möglichkeit, unterschiedliche Komponentenausprägungen für unterschiedliche Zwecke zu verwenden. Nachteilig ist jedoch die Notwendigkeit zur Spezialisierung durch die Anwendung, die die Komponentenausprägungen nutzen möchte. Diese muß die Spezialisierungsinformationen selbst verwalten.

9.1.4 Komponententypen

Bevor mit der Definition des Begriffs Komponententyp begonnen werden kann, muß zunächst die Bedeutung des Begriffs Typ geklärt werden, da er in einer Vielzahl von homonymen und synonymen Verwendungen auftaucht. Für diese Arbeit wird dazu die Definition aus [Goos99] verwendet. Weiterführende Überlegungen sind in [HeRe99] zu finden. Unter einem Typ wird ein Baumuster für Objekte verstanden. In objektorientierten Sprachen werden diese Baumuster Klassen genannt. Überträgt man diese Vorstellung eines Baumusters auf Komponenten, so wird bei Komponenten dieses Baumuster durch die ein- und ausgehenden Schnittstellen repräsentiert.

Definition 23 **Komponententyp**
Der Typ einer Komponente wird durch ihre ein- und ausgehenden Schnittstellen festgelegt.

Durch die Einführung von Komponententypen wird eine Indirektionsebene geschaffen. Es kann auf die Referenzierung einer bestimmten Komponente verzichtet werden. Dies kann für eine Vielzahl von Zwecken genutzt werden, wie beispielsweise die transparente Bereitstellung eines Dienstes durch unterschiedliche Komponenten vom gleichen Typ. Dargestellt werden Komponententypen durch das in Abbildung 76 dargestellte Symbol. Zur Unterscheidung von Komponenten werden Komponententypen mit einer unterbrochenen Begrenzung gezeichnet.

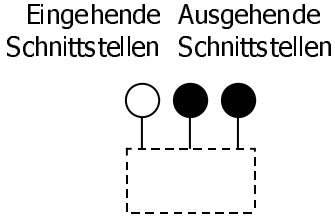


Abbildung 76: Symbol für Komponententyp

Eine Komponente ist vom gleichen Typ wie eine andere Komponente, wenn sie die gleichen Schnittstellen unterstützt. Die Komponententypen befinden sich in einer hierarchischen Ordnung aus Ober- und Untertypen. In Anlehnung an die Definition in [Goos99] wird ein Komponentenuntertyp folgendermaßen definiert:

Definition 24 **Komponentenuntertyp**
Ein Komponententyp A ist Untertyp eines Komponententyps B, wenn Komponententyp A eine Obermenge der Schnittstellen von Komponententyp B unterstützt.

Die Eigenschaften von Komponententypen sollen an Hand eines Beispieles veranschaulicht werden. Betrachtet werden zwei Komponententypen k1 und k2. Der Komponententyp k1 verfügt über die Schnittstelle s1 und der Komponententyp k2 über die Schnittstellen s1 und s2. Der Komponententyp k2 ist daher Untertyp von k1.

Zusammenfassend können jetzt die Bestandteile einer Komponente dargestellt werden. Eine Übersicht gibt Abbildung 77. Komponenten sind Komponententypen zugeordnet und können Ausprägungen bilden. Komponenten weisen drei Konzepte auf: Schnittstellen, Introspektions- und Spezialisierungsmechanismen. Hierdurch können sie leichter in neue Anwendungskontexte übertragen werden. Schnittstellen können in eingehender und ausgehender Form auftreten. Introspektionsmechanismen geben Auskunft über die von einer Komponente angebotenen Dienste. Mit Hilfe der Spezialisierungsmechanismen können die von einer Komponente erbrachten Dienste angepaßt werden. Dies erfordert keine Kenntnis der Implementierungsdetails der Komponente, da sowohl Implementierungs- als auch Spezialisierungsmechanismen über Schnittstellen gekapselt sind.

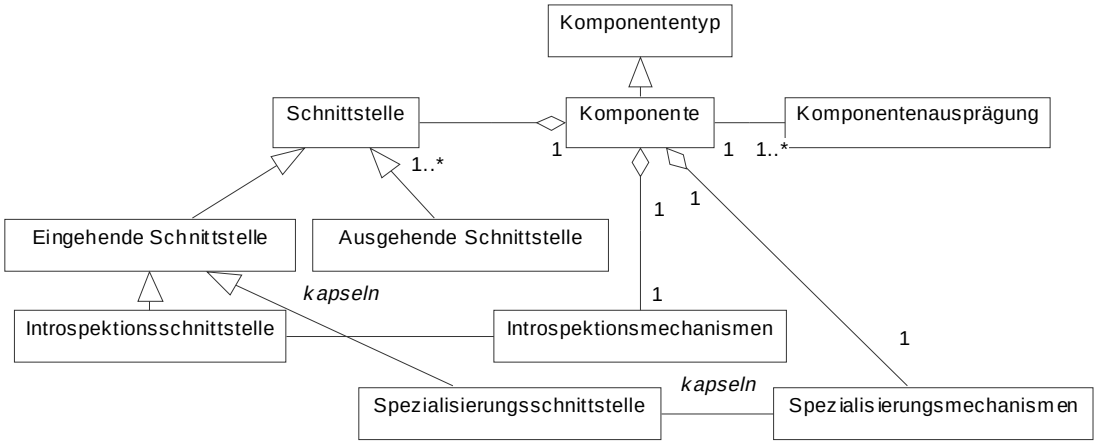


Abbildung 77: Komponente

9.1.5 Anwendungsentwicklung mit Komponenten

Die Entwicklung einer Anwendung mit Hilfe von Komponenten ist zweigeteilt. Außer dem Entwickler der Anwendung, also dem Komponentenanwender, gibt es noch den Komponentenentwickler. Dieser muß mit dem Anwendungsentwickler in keinem direkten Kontakt stehen.

Der Ablauf einer Anwendungsentwicklung ist in Abbildung 78 dargestellt. Der Komponentenanwender bezieht die Komponente entweder direkt oder indirekt vom Komponentenentwickler. Als Spezifikation kann dabei der Komponententyp dienen. Als Vermittlungsebene kommen dabei Zwischenhändler, Tausch, Märkte usw. in Frage. Der Komponentenanwender paßt mit Spezialisierungsinformationen die Komponente für seine Anwendung an und fügt sie der Anwendung hinzu. Die fertige Anwendung kann schließlich vom Anwender genutzt werden.

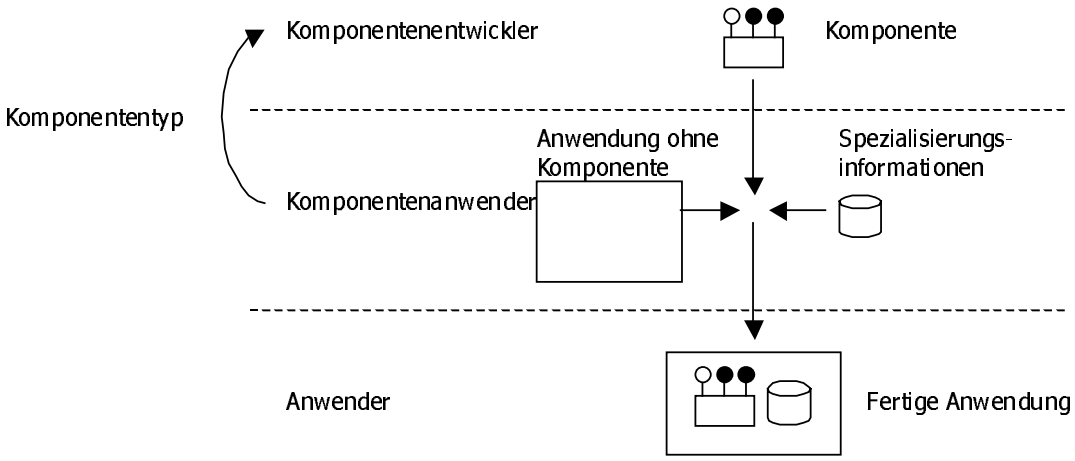


Abbildung 78: Anwendungsentwicklung mit Komponenten

9.1.6 Bewertung

Introspektions- und Spezialisierungsmechanismen sind der Schlüssel zur Konfigurierbarkeit der von Komponenten bereitgestellten Dienste. Durch sie ist es möglich, zur Laufzeit Parameter zu

verändern, die die Eigenschaften von Diensten festlegen. Dies kann für jede Komponentenausprägung separat erfolgen, so daß eine individuelle Anpassung von Diensten möglich ist. Daher erfüllen Komponenten die Anforderung der Dienstekonfigurierbarkeit.

Dies wird an einer Komponente veranschaulicht, die einen Dienst anbietet, der die bedingten Verzweigungen im Rahmen eines Workflows unterstützt. Die bedingte Verzweigung wird abhängig von einem Entscheidungsparameter und einem Entscheidungswert getroffen werden. Übersteigt der Wert des Entscheidungsparameters den Entscheidungswert, wird eine anzugebende Schnittstelle aktiviert, andernfalls eine andere. Der Name des Entscheidungsparameters sowie des Entscheidungswerts kann über die Spezialisierungsmechanismen festgelegt werden. Beispielsweise ist anfänglich der Parametername „Auftragswert“ und der Entscheidungswert 5000. Durch Einsatz der Spezialisierungsmechanismen kann der Entscheidungswert auf 6000 verändert werden.

9.2 Komponentenorientierte Rahmenwerke

Ziel komponentenorientierter Rahmenwerke ist es, Komponenten in unterschiedlichen Anwendungen gleichzeitig verwenden zu können. Hierzu führen sie Mechanismen ein, die Ausprägungen der Komponenten transparent nutzbar machen. Wichtigstes Mittel ist das Lebenszyklusmanagement mit Hilfe der sogenannten *Ausprägungsverwaltung*. Sie unterstützt die Erzeugung, Verwaltung und Zerstörung von Komponentenausprägungen. Eine wichtige unterstützende Rolle nimmt dabei die *Registratur* ein, die ein Verzeichnis aller Komponenten darstellt, von denen Ausprägungen gebildet werden können. Die *Transparenzschicht* ist ein weiterer wichtiger Bestandteil komponentenorientierter Rahmenwerke. Durch sie ist eine orts- und implementierungstransparente Nutzung von Komponentenausprägungen in komponentenorientierten Rahmenwerken möglich.

Darüber hinaus besitzen komponentenorientierte Rahmenwerke Standarddienste, die häufig benötigte Funktionalität bereitstellen. Hierzu gehören *Kooperationsdienste,* die unterschiedliche Formen der Kooperation von Komponenten unterstützen. Diese reichen vom einfachen Nachrichtenaustausch bis zu Transaktionen. *Integrationsdienste* erlauben zudem die Integration externer Informationsquellen. Schließlich existieren vermehrt auch komponentenorientierte Mechanismen zur WWW-Integration, bei der auf der Basis von Komponenten HTML erzeugt wird. Dargestellt sind die Bestandteile eines komponentenorientierten Rahmenwerks in Abbildung 79.

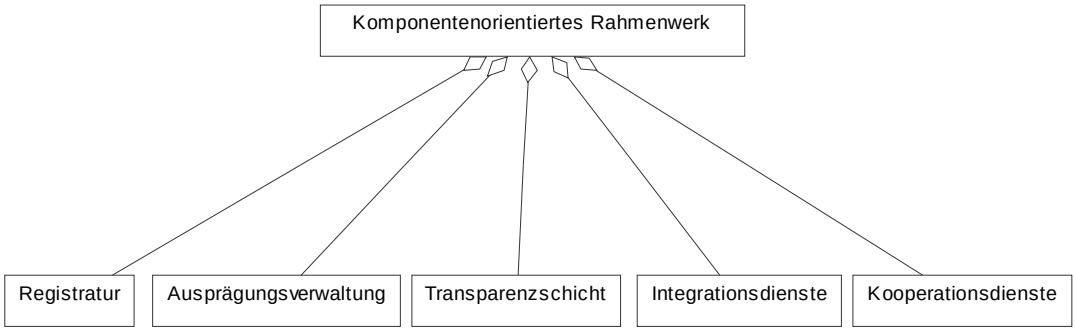


Abbildung 79: Bestandteile komponentenorientierter Rahmenwerke

9.2.1 Registratur

Die Registratur enthält die Informationen über die Komponenten eines komponentenorientierten Rahmenwerks. Die Komponenten befinden sich außerhalb der Registratur. Die Registratur teilt die Komponenten nach Komponententypen ein. In der Registratur sind jedem

Komponententyp eine oder mehrere Komponenten- und Schnittstellenbeschreibungen zugeordnet, wie in Abbildung 80 dargestellt.

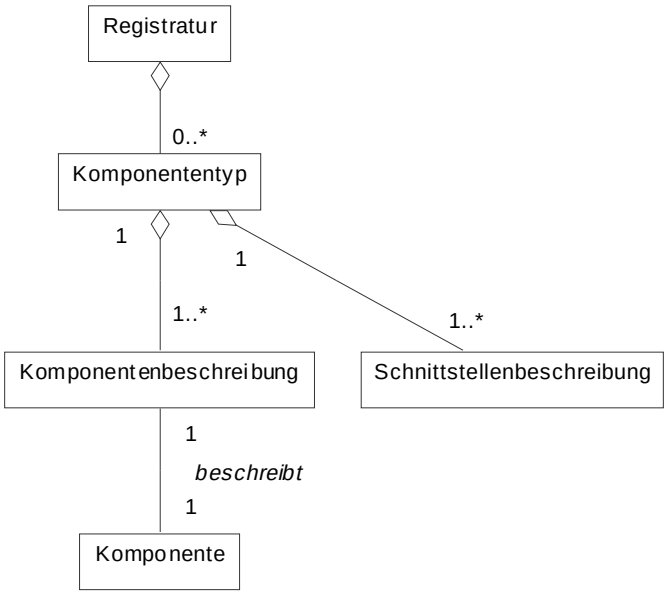


Abbildung 80: Registratur

Jeder Komponentenbeschreibung ist eine Komponente zugeordnet. In der Komponentenbeschreibung werden die Informationen wie der physikalische Ort der Komponente eingetragen. Die Schnittstellenbeschreibungen dienen zur Beschreibung der den Komponententyp konstituierenden Schnittstellen. Sind mehrere Komponentenbeschreibungen und damit Komponenten zu einem Komponententyp vorhanden, so trifft eine Auswahlfunktion die Entscheidung, von welcher Komponente unter welchen Bedingungen eine Ausprägung gebildet wird. Ein Kriterium kann beispielsweise die Auslastung der Rechner sein, auf denen sich die Komponenten befinden. Die Aufnahme von Komponententypen und Komponentenbeschreibungen in die Registratur kann zur Laufzeit erfolgen.

Nun wird ein Beispiel für eine Registratur angegeben. Sie ist in Abbildung 81 wiedergegeben. In der Registratur sind zwei Komponententypen, k1 und k2 genannt, verzeichnet. Dem Komponententyp k1 ist die Schnittstellenbeschreibung s1, dem Komponententyp k2 die Schnittstellenbeschreibungen s1 und s2 zugeordnet. Unter den Komponententypen k1 und k2 sind drei Komponenten verzeichnet, die durch Komponentenbeschreibungen repräsentiert sind. Unter dem Komponententyp k1 ist die Komponente $k1_1$ und unter dem Komponententyp k2 sind die Komponenten $k2_1$ und $k2_2$ verzeichnet. Der Mechanismus für die Auswahl zwischen den Komponenten $k2_1$ und $k2_2$ ist nicht dargestellt. Falls sich die Komponenten $k2_1$ und $k2_2$ auf verschiedenen Rechnern befinden, bildet dieser Mechanismus immer von der Komponente Ausprägungen, die sich auf dem weniger ausgelasteten Rechner befindet.

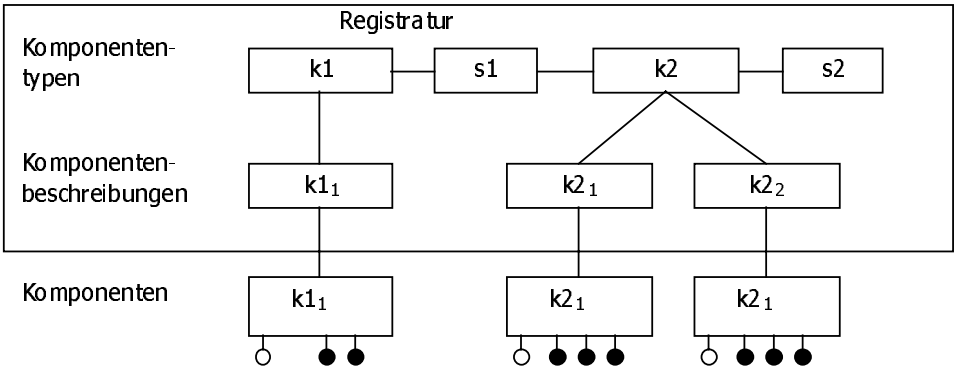


Abbildung 81: Beispiel für eine Registratur

9.2.2 Ausprägungsverwaltung

Die Ausprägungsverwaltung stellt einen Indirektionsmechanismus zwischen Komponententypen und Komponenten dar und führt das Lebenszyklusmanagement der Komponenten durch.

Die Ausprägungsverwaltung stellt Ausprägungen nicht unter Angabe einer Komponente, sondern eines Komponententyps bereit. Anwendungen geben daher immer einen Komponententyp, und nicht eine konkrete Komponente an, wenn sie Ausprägungen anfordern. Auf diese Weise brauchen Anforderungen keine konkrete Komponente zu kennen.

Wird von einer Anwendung unter Angabe des Komponententyps eine Ausprägung angefordert, so bestimmt die Ausprägungsverwaltung mit Hilfe der Informationen in der Registratur eine dem Komponententyp entsprechende Komponente und bildet von dieser eine Ausprägung. Durch die Verwendung von Komponententypen können für die Bereitstellung von Ausprägungen unterschiedliche Komponenten verwendet werden, die sich zudem an unterschiedlichen Orten befinden können.

Um den transparenten Zugriff auf die so gebildeten Ausprägungen sicherzustellen, erhalten Anwendungen keinen direkten Zugriff mehr auf die Komponentenausprägungen. Statt dessen wird ein Stellvertretermechanismus verwendet. Die Transparenzschicht übernimmt die Überbrückung zwischen dem Stellvertreter der Komponentenausprägung und der eigentlichen Komponentenausprägung. Auf diese Weise wird die Grundlage für eine transparente Nutzung der Komponentenausprägungen über Rechner- und Systemgrenzen hinweg gelegt.

Veranschaulicht ist dies an Hand der Nutzung einer Ausprägung der Komponente $k1_1$ vom Typ k1, dargestellt in Abbildung 82.

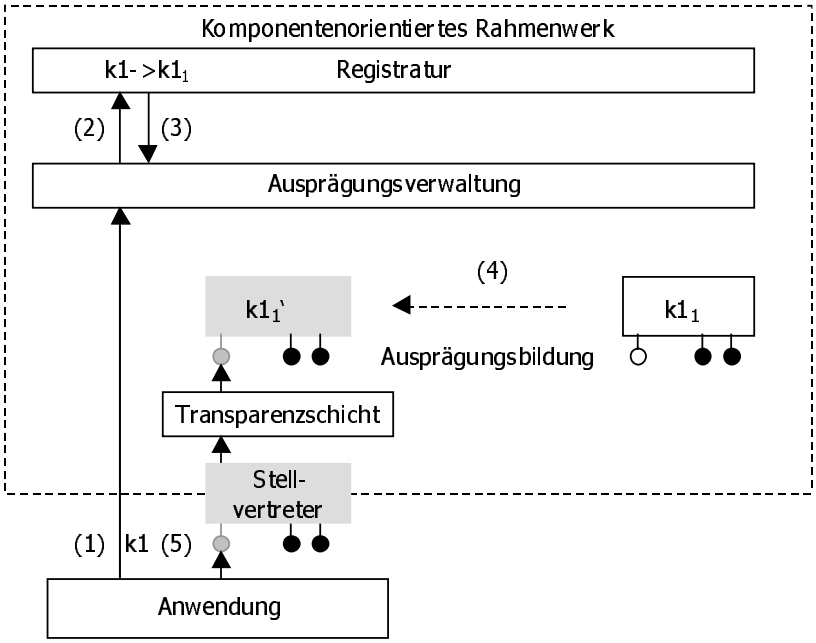


Abbildung 82: Erzeugung von Komponentenausprägungen

Die Komponente $k1_1$ ist in der Registratur des Rahmenwerks unter dem Typ k1 eingetragen. In Schritt (1) fordert die Anwendung unter Angabe des Komponententyps k1 eine Ausprägung an. Die Ausprägungsverwaltung nimmt die Anforderung entgegen und sucht in der Registratur nach einer geeigneten Komponente (2). Die Auswertung der Registratur ergibt, daß die Komponente $k1_1$ dem Komponententyp k1 entspricht. Ebenfalls werden alle Informationen, die zur Erzeugung einer Ausprägung der Komponente $k1_1$ notwendig sind, zurückgegeben (3). Mit diesen Informationen erzeugt die Ausprägungsverwaltung die Ausprägung $k1_1$`, sowie einen passenden Stellvertreter (4). Schließlich erhält die Anwendung den Stellvertreter und kann so die Komponentenausprägung nutzen (5). Die Transparenzschicht sorgt dabei für die Verbindung zwischen Stellvertreter und der Ausprägung $k1_1$`.

Die Stellvertretermechanismen und die Transparenzschicht werden in den folgenden Abbildungen nur dann dargestellt, wenn dies zum Verständnis notwendig ist.

9.2.3 Ressourcenoptimierung

Komponentenorientierte Rahmenwerke lassen sich auch als Ressourcenverwalter auffassen, mit den Komponentenausprägungen als Ressourcen. Daher stellt sich die Frage, wie die Nutzung dieser Ressourcen optimiert werden kann. Kriterien sind beispielsweise der von den Komponentenausprägungen verbrauchte Speicherplatz, die genutzten Datenbankverbindungen etc.

Die durch komponentenorientierte Rahmenwerke durchgeführte Optimierung nutzt den Umstand aus, daß die Komponentenausprägungen für die Anwendungen nicht direkt, sondern nur über einen Stellvertreter sichtbar sind. Daher brauchen nur so viele laufende Komponentenausprägungen erzeugt werden, wie aktuell benötigt werden. Dementsprechend lassen sich der benötigte Speicherplatz usw. reduzieren.

Für zustandslose Komponenten ist die Optimierung in Abbildung 83 veranschaulicht. Drei Anwendungen (1-3) haben eine Ausprägung vom Komponententyp k1 angefordert. Als Folge davon haben sie von der Ausprägungsverwaltung (nicht dargestellt) je einen Stellvertreter erhal-

ten. Durch die Stellvertreter kann die Ausprägungsverwaltung feststellen, welche der Anwendungen die angeforderte Komponentenausprägung effektiv benötigt. Es wird angenommen, daß die Anwendungen 1 und 3 ihre Komponentenausprägungen sofort nutzen wollen, Anwendung 2 jedoch nicht. Daher kann das komponentenorientierte Rahmenwerk auf die Erzeugung einer Komponentenausprägung für Anwendung 2 zunächst verzichten. Für Anwendung 2 ist dies jedoch nicht sichtbar, da sie nur den Stellvertreter der Komponente sieht.

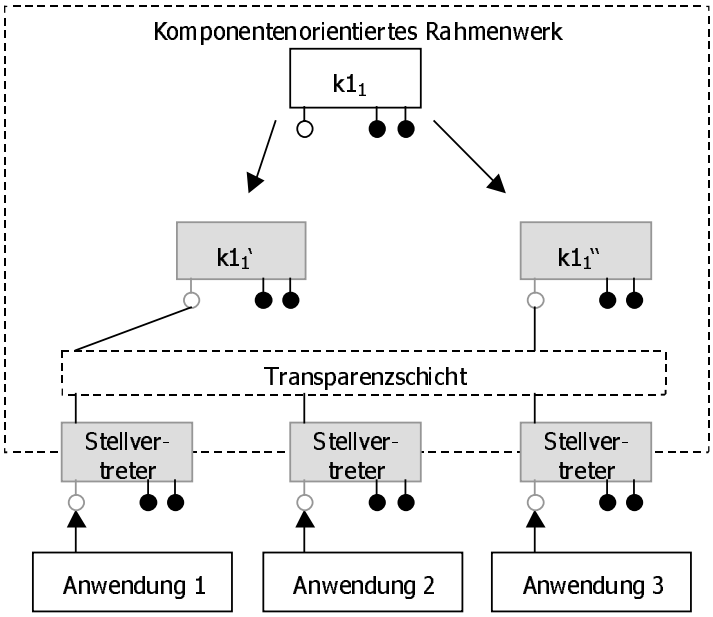


Abbildung 83: Optimierte Nutzung zustandsloser Komponentenausprägungen (1)

Es wird angenommen, daß Anwendung 1 ihre Komponentenausprägung zeitweise nicht benötigt. Statt dessen soll Anwendung 2 seine Komponentenausprägung benötigen. In diesem Fall kann die Komponentenausprägung k1‘ von Anwendung 2 genutzt werden. Dargestellt ist dies in Abbildung 84. Nach Abschluß der Nutzung von k1‘ durch Anwendung 2 kann sie wiederum durch Anwendung 1 oder eine andere Anwendung genutzt werden.

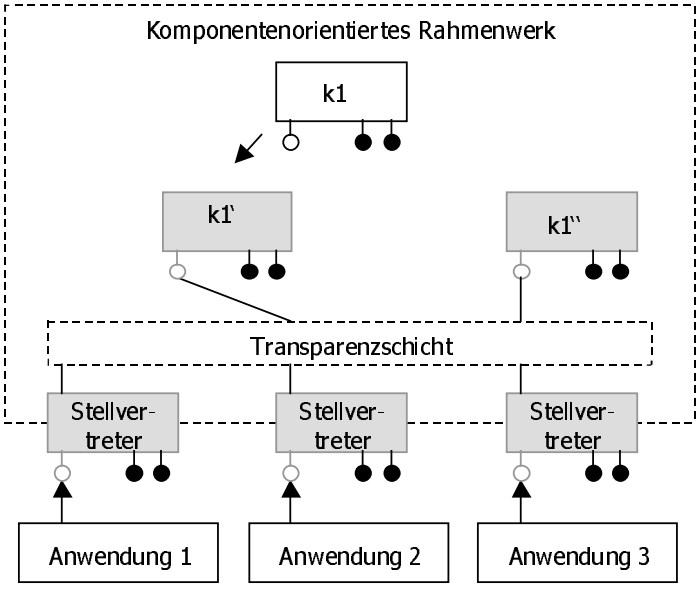


Abbildung 84: Optimierte Nutzung zustandsloser Komponentenausprägungen (2)

Die hier beschriebene Optimierung der Ressourcennutzung ist für zustandslose Komponenten direkt anwendbar. Für zustandsbehaftete Komponenten ist eine Erweiterung notwendig. Hierbei

wird der Zustand der Komponentenausprägungen persistent zwischengespeichert. Ebenfalls sind die Überlegungen auf andere Ressourcen wie beispielsweise Datenbankverbindungen ausdehnbar. Hierzu werden die Datenbankanforderungen mehrerer Komponentenausprägungen zusammengefaßt und auf diese Weise die Zahl der gleichzeitig geöffneten Datenbankverbindungen reduziert.

9.2.4 Bewertung

Nun wird untersucht, welche der nicht-funktionalen Anforderungen durch komponentenorientierte Rahmenwerke erfüllt werden. Zur Veranschaulichung dient Abbildung 85.

In komponentenorientierten Rahmenwerken ist es möglich, zur Laufzeit neue Komponententypen und Komponenten einzubringen. Durch die Aufnahme zusätzlicher Komponententypen ist es möglich, die Menge der in einem komponentenorientierten Rahmenwerk angebotenen Dienste zu erweitern. Durch die Aufnahme zusätzlicher Komponenten kann die Menge der Ressourcen, die diese Dienste erbringen, erweitert werden. Daher werden die Anforderung der Dienste- und Ressourcenerweiterbarkeit erfüllt.

Die Aufnahme eines neuen Komponententyps wird an Hand der Aufnahme des Komponententyps „Embargoprüfung“ veranschaulicht. Der neue Komponententyp ist oe. Die dazugehörige Komponente oe_1. Zunächst wird in die Registratur der Komponententyp oe eingetragen. Der nächste Schritt ist die Schaffung einer Komponentenbeschreibung. In diese werden die Informationen zur Komponente oe_1 eingetragen.

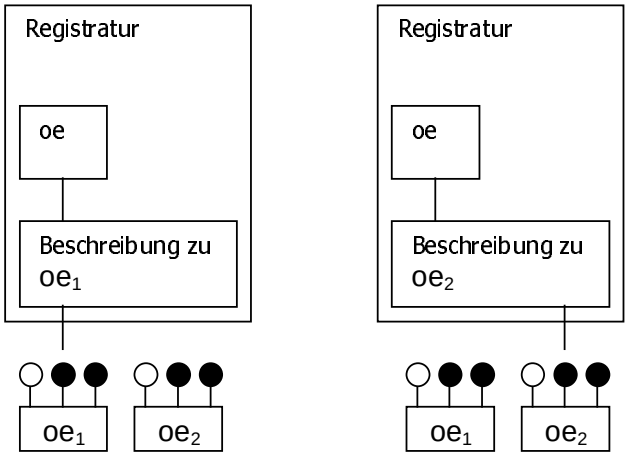


Abbildung 85: Aufnahme neuer Komponenten

Durch die Registratur wird auch die flexible Zuweisung von Ressourcen in komponentenorientierten Rahmenwerken erreicht. So können Komponenten für die Unterstützung eines bestimmten Komponententyps transparent gewechselt werden. Hierzu ist nur die Änderung der Zuweisung, die in der Komponentenbeschreibung vorgenommen wird, erforderlich. Daher wird auch die Anforderung der Ressourcenkonfigurierbarkeit erfüllt. Zur Veranschaulichung dient der Wechsel der Komponente oe_1 gegen die Komponente oe_2. Sie wird durch den Austausch der Komponentenbeschreibung erreicht wird.

Komponentenorientierte Rahmenwerke erfüllen die Anforderungen der Dienste- und Ressourcenerweiterbarkeit sowie der Ressourcenkonfigurierbarkeit. Sie erfüllen jedoch nicht die Anforderungen der unabhängigen Anwendungserweiterbarkeit und –konfigurierbarkeit. Es fehlt eine Architektur, die es ermöglicht, Anwendungen dynamisch um zusätzliche Dienste zu erweitern bzw. die Verknüpfung der Dienste innerhalb einer Anwendung zur Laufzeit anzupassen.

9.3 Komposite Anwendungen

Um eine Anwendungsarchitektur bereitzustellen, die die Anforderungen der unabhängigen Anwendungserweiterbarkeit und –konfigurierbarkeit erfüllt, wird das Konzept der kompositen Anwendung [Schm97a], [Schm97c], [ScAs98d] eingeführt. Die Herausforderung bei der Entwicklung des Konzepts kompositer Anwendungen liegt nicht nur in der Erfüllung der gerade genannten Anforderungen, sondern der gleichzeitigen Erfüllung sämtlicher nicht-funktionaler Anforderungen. Es sollen also auch die Anforderungen der Dienste- und Ressourcenkonfigurierbarkeit sowie Dienste- und Ressourcenerweiterbarkeit durch die kompositen Anforderungen erfüllt werden. Das Konzept der kompositen Anwendung muß somit verträglich zu den bisher vorgestellten Konzepten der Komponente und des komponentenorientierten Rahmenwerks gestaltet werden.

Die Definition in Abschnitt 2.4 sieht eine Anwendung als Verknüpfung von Diensten an. Bei Komponenten werden Dienste über Schnittstellen bereitgestellt. Daher geschieht die Verknüpfung von Diensten bei Komponenten über die Verknüpfung von Schnittstellen.

Grundidee der kompositen Anwendung ist es, eine Anwendung nicht mehr durch eine statische, sondern eine dynamische Verknüpfung von Diensten zu schaffen. Diese dynamische Verknüpfung wird mit Hilfe der Spezialisierungsmechanismen der Komponenten durchgeführt, die Verknüpfungsinformationen in die Komponentenausprägungen einbringen. Die Verknüpfungsinformationen bestehen aus der Angabe eines Zielkomponententyps und der zu verknüpfenden Schnittstelle, wie in Abbildung 86 dargestellt. Die Schnittstelle im Zielkomponententyp ist dabei vom gleichen Typ wie die ausgehende Schnittstelle, die deswegen nicht angegeben wird. Prinzipiell wäre auch die Verknüpfung unterschiedlicher Schnittstellen denkbar. Dann wäre aber ein Abbildungsmechanismus notwendig, der die beiden Schnittstellen zueinander kompatibel macht.

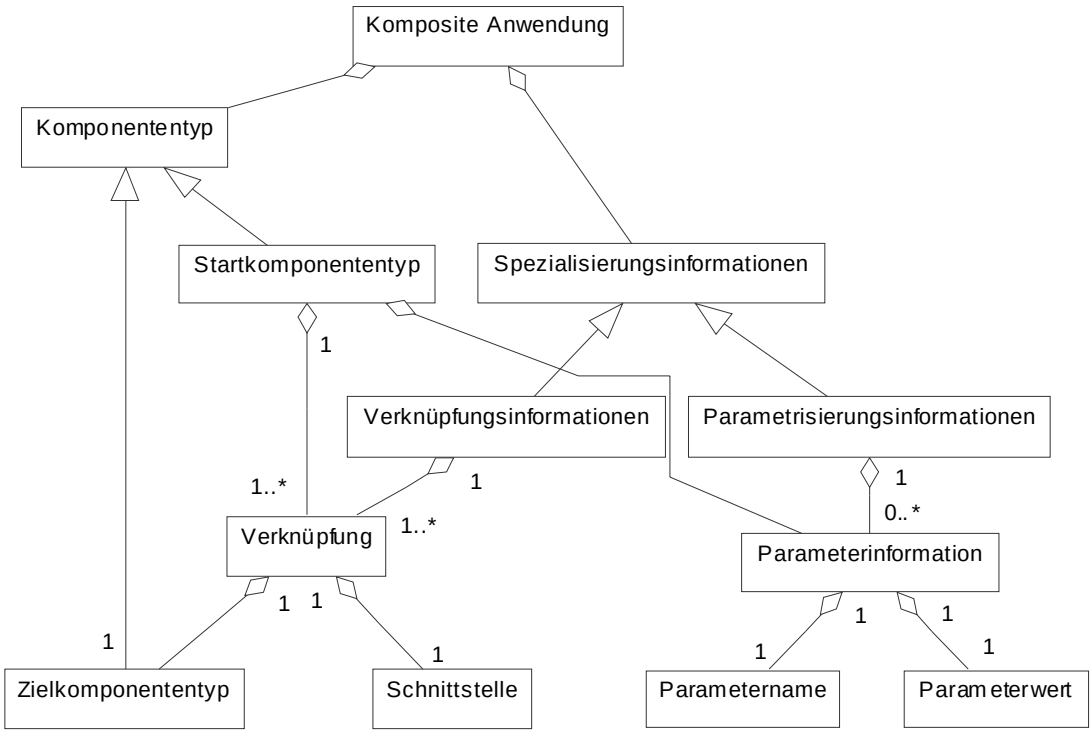


Abbildung 86: Struktur kompositer Anwendungen

In den Verknüpfungsinformationen werden Komponententypen und nicht Komponenten verwendet, um die durch die Ausprägungsverwaltung durchgeführte Indirektion von Komponententypen auf Komponenten nutzen zu können. Auf diese Weise wird die Verträglichkeit mit komponentenorientierten Rahmenwerken gesichert und so die Einhaltung der Anforderungen der Dienst- und Ressourcenerweiterbarkeit sowie der Dienstekonfigurierbarkeit gewährleistet. So können unterschiedliche Komponenten genutzt werden, ohne daß dies in der kompositen Anwendung sichtbar wird. Die Menge der zur Bildung von kompositen Anwendungen bereitstehenden Komponententypen kann zudem vergrößert werden.

Neben Verknüpfungsinformationen enthält eine komposite Anwendung auch Parametrisierungsinformationen, mit der die Anpassung von Diensten durchgeführt werden kann. Dies geschieht mit den in Abschnitt 9.1.2 vorgestellten Spezialisierungsmechanismen. Daher wird auch die Anforderung der Dienstekonfigurierbarkeit erfüllt werden, da die verwendeten Dienste auf diesem Wege angepaßt werden können.

Ein Beispiel für eine komposite Anwendung ist in Abbildung 87 dargestellt. Die komposite Anwendung besteht aus zwei Komponententypen o1 und k1. Die Spezialisierungsinformationen des Komponententyps o1 bestehen nur aus Verknüpfungsinformationen. Sie enthalten für Komponententyp o1 eine Verknüpfung mit Komponententyp k1 über die Schnittstelle s3. Die Spezialisierungsinformationen für die Komponente k1 enthalten Verknüpfungsinformationen und Parametrisierungsinformationen. Die Verknüpfungsinformationen enthalten eine Verknüpfung mit dem finalen Komponententyp e. Die Parametrisierungsinformationen enthalten die Festlegung des Werts 5000 für den Parameter Auftragswert.

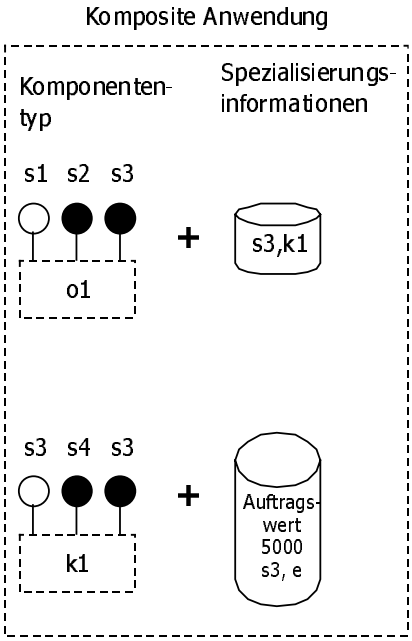


Abbildung 87: Komposite Anwendung

Zur Anpassung einer kompositen Anwendung müssen nur die Verknüpfungsinformationen und Parametrisierungsinformationen angepaßt werden, was durch die Verwendung der in Abschnitt 9.3.1 beschriebenen Spezialisierungsmechanismen zur Laufzeit geschehen kann. Dies geschieht wiederum ohne Neuübersetzung der Komponenten und ebenfalls ohne Interpretation. D.h. die Spezialisierungsinformationen sind ohne Interpretationsvorgänge wirksam.

Eine Ausprägung einer kompositen Anwendung ist eine Menge von Komponentenausprägungen, die entsprechend dem, durch die komposite Anwendung vorgegebenen Muster zusam-

menarbeiten. Die Bildung von Ausprägungen einer kompositen Anwendung wird an Hand von Abbildung 88 verdeutlicht.

Betrachtet wird eine komposite Anwendung, die aus zwei Komponententypen o1 und k1 besteht. Für diese Komponententypen sind Spezialisierungsinformationen mit Verknüpfungs- und Parametrisierungsinformationen angegeben. Für den Komponententyp o1 ist in den Verknüpfungsinformationen eine Verknüpfung mit dem Komponententyp k1 über die Schnittstelle s3 eingetragen. Die Verknüpfungsinformationen für den Komponententyp k1 enthalten eine Verknüpfung mit dem Komponententyp e, der die Ausführung der kompositen Anwendung beendet. Die Parametrisierungsinformationen für den Komponententyp k1 enthalten den Wert 5000 für den Parameter Auftragswert. Es wird außerdem angenommen, daß die Komponenten $o1_1$ und $k1_1$ in der Registratur für Komponententyp o1 und k1 eingetragen sind. Von den Komponenten wurden unter Verwendung der Spezialisierungsinformationen die Komponentenausprägungen $o1_1$' und $k1_1$' gebildet. Sie zusammen bilden eine Ausprägung der kompositen Anwendung.

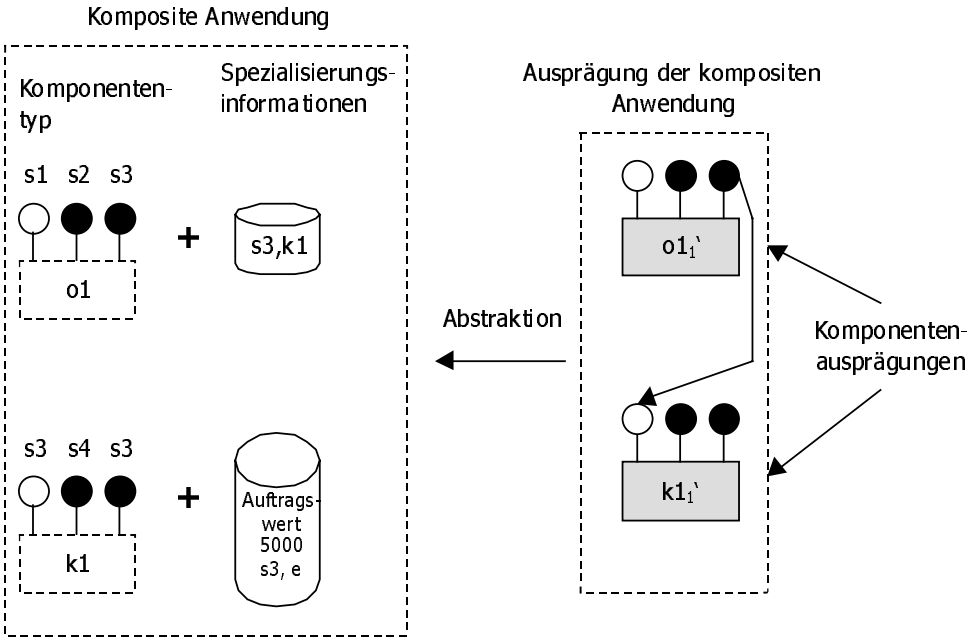


Abbildung 88: Ausprägung einer kompositen Anwendung

Werden von beiden Komponenten neue Ausprägungen gebildet, beispielsweise als Komponentenausprägungen o1" und k1", so wird eine weitere, unabhängige Ausprägung der kompositen Anwendung gebildet. Ebenso lassen sich auf der Basis unterschiedlicher Spezialisierungsinformationen unterschiedliche komposite Anwendungen schaffen, die auf unterschiedlichen Ausprägungen derselben Komponenten beruhen.

Zusammenfassend kann nun folgende Definition einer kompositen Anwendung gegeben werden:

Definition 25 **Komposite Anwendung**
Eine komposite Anwendung abstrahiert vom Zusammenspiel von Komponentenausprägungen in Form von Komponententypen, denen Verknüpfungs- und Parametrisierungsinformationen zugeordnet sind. Die Verknüpfungsinformationen verknüpfen die Dienste der durch Komponententypen spezifizierten Komponenten. Die Parametrisierungsinformationen passen die Dienste der durch Komponententypen spezifizierten Komponenten an.

9.3.1 Neue Formen der Spezialisierung in kompositen Anwendungen

Für die Bildung kompositer Anwendungen sind die in Abschnitt 9.1.2 vorgestellten Spezialisierungstechniken prinzipiell tauglich. Dennoch wäre es sehr ungünstig, diese für komposite Anwendungen zu verwenden. Die generelle Spezialisierung erfordert für jede komposite Anwendung eine Kopie der Komponente. Die individuelle Spezialisierung erfordert zur Anpassung der Spezialisierungsinformationen eine Veränderung der Anwendung, die die Spezialisierung durchführt. Daher wird durch die individuelle Spezialisierung die Anforderung der Anwendungskonfigurierbarkeit verletzt. Die hier neu eingeführte anwendungsdokument- und registraturbasierte Spezialisierung vermeidet die Nachteile der generellen und individuellen Spezialisierung. Sie ermöglichen sowohl die dynamische Zuordnung von Spezialisierungsinformationen zu einer Komponente, als auch die dynamische Anpassung der Spezialisierungsinformationen.

Unterstützt werden beide Verfahren durch initiale und finale Komponenten. Initiale Komponenten werden als Ausgangspunkt für eine komposite Anwendung auf der Basis eines Anwendungsdokumentes benötigt und tragen keine Anwendungsfunktionalität. Ähnliches gilt für finale Komponenten, die die Ausführung der kompositen Anwendung zu einem definierten Ende führen. Initiale Komponenten werden unter dem Komponententyp i, finale Komponenten unter dem Komponententyp e erfaßt.

9.3.1.1 Anwendungsdokumentbasierte Spezialisierung

Die anwendungsdokumentbasierte Spezialisierung faßt die Spezialisierungsinformationen in einem sogenannten Anwendungsdokument zusammen, wie in Abbildung 89 dargestellt. Das Anwendungsdokument enthält die Spezialisierungsinformationen aller, an der kompositen Anwendung beteiligten Komponententypen. Dieses Anwendungsdokument wird über eine spezielle Schnittstelle s_a von Komponente zu Komponente weitergegeben. Hierdurch entfällt die Notwendigkeit, die Spezialisierungsschnittstelle s_{sp} zu benutzen.

Die Spezialisierungsinformationen in dem Anwendungsdokument haben folgende Syntax: Für jeden Komponententyp sind die Verknüpfungs- und Parametrisierungsinformationen angegeben. Die Verknüpfungsinformationen bestehen aus der Schnittstelle, von der die Verknüpfung ausgehen soll, und den Zielkomponententypen. Parametrisierungsinformationen bestehen aus der Angabe des Parameters und seinem Wert. Unterschiedliche Verknüpfungen und Parametrisierungen sind jeweils durch einen Strichpunkt voneinander getrennt.

In Abbildung 89 ist ein Beispiel für ein Anwendungsdokument und dazugehörigen Komponenten sowie Registratureinträge dargestellt. Die Spezialisierungsinformationen des Komponententyps o1 bestehen nur aus Verknüpfungsinformationen. Die Schnittstelle s_a ist mit dem Zielkomponententyp k1 verknüpft. Die Spezialisierungsinformationen des Komponententyps k1 enthalten Verknüpfungs- und Parametrisierungsinformationen. Die Verknüpfungsinformationen enthalten die Verknüpfung mit der Endkomponente e. Die Parametrisierungsinforma-

tionen bestehen aus der Angabe des Entscheidungsparameters „Auftragswert“ und dem Wert 5000.

Zur Umsetzung der kompositen Anwendung stehen die Komponenten $o1_1$ sowie $k1_1$ bereit. Außerdem werden vom komponentenorientierten Rahmenwerk die initiale Komponente i_1 und die finale Komponente e_1 bereitgestellt. Alle Komponenten sind ihren Komponententypen entsprechend in die Registratur eingetragen. In den späteren Abbildungen wird die Registratur nicht mehr dargestellt.

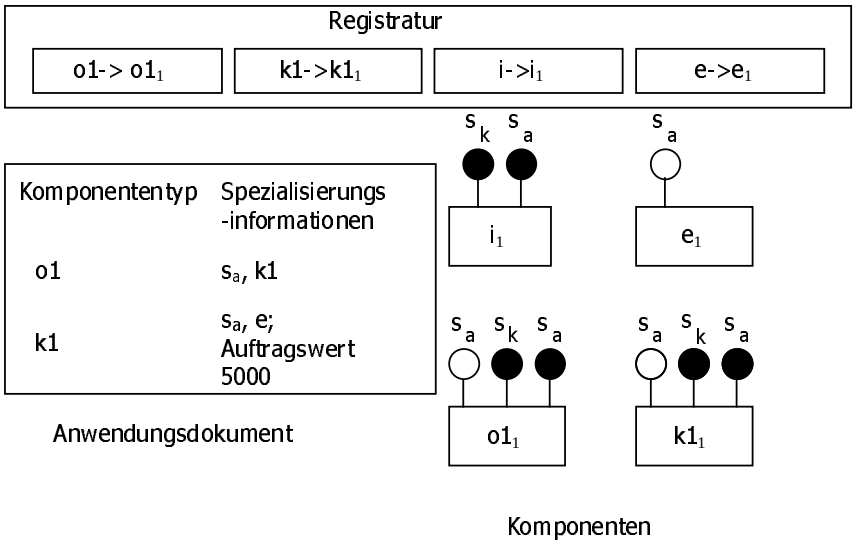


Abbildung 89: Anwendungsdokumentbasierte Spezialisierung

Die Ausführung von kompositen Anwendungen mit Hilfe der anwendungsdokumentbasierten Spezialisierung geschieht durch das zyklische Durchlaufen von vier Schritten. Durch die Auswertung der Verknüpfungsinformationen wird zunächst der Komponententyp identifiziert. Im zweiten Schritt wird über die ausgehende Schnittstelle s_k von der Ausprägungsverwaltung eine Komponentenausprägung unter Angabe des Komponententyps angefordert. Im dritten Schritt wird das Anwendungsdokument an die Komponentenausprägung übergeben, die vom komponentenorientierten Rahmenwerk zurückgeliefert wurde. Dies geschieht durch die Schnittstelle s_a. Diese Schnittstelle ist bei allen Komponenten sowohl in ein- und ausgehender Form vorhanden, der in Abbildung 71 eingeführten Notation folgend, als Kreis bzw. Kreisfläche dargestellt. Eine Ausnahme bilden lediglich initiale und finale Komponenten. Sie besitzen die Schnittstelle nur in aus- bzw. eingehender Form. Im vierten Schritt wird die Komponentenausprägung unter Nutzung der Spezialisierungsinformationen im Anwendungsdokument spezialisiert.

Dieses Vorgehen wird durch Abbildung 90 verdeutlicht. Grundlage ist das Anwendungsdokument aus Abbildung 89. Die Ausprägung i_1' der Komponente i_1 wertet das Anwendungsdokument aus (1) und fordert von der Ausprägungsverwaltung eine Ausprägung von Komponententyp o1 an (2). Die Ausprägungsverwaltung erzeugt daraufhin eine Ausprägung der Komponente $o1_1$ und gibt dieses an i_1' zurück. Daraufhin leitet i_1' das Anwendungsdokument an die Ausprägung $o1_1'$ weiter (3). Den Abschluß bildet die Spezialisierung der Komponentenausprägung $o1_1'$ (4). Dies geschieht, indem $o1_1'$ das Anwendungsdokument auswertet, also ohne äußere Einflußnahme.

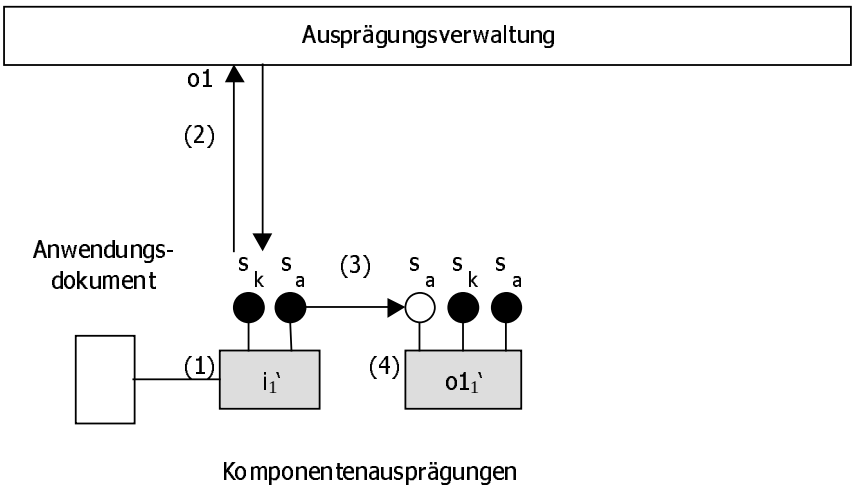


Abbildung 90: Ausführung einer kompositen Anwendung (1)

Der nächste Schritt der Ausführung der kompositen Anwendung ist in Abbildung 91 dargestellt. Ausgangspunkt ist die spezialisierte Komponentenausprägung $o1_1$‘. In ihr ist die Verknüpfung mit dem Komponententyp k1 abgelegt. Nach Auswertung des Anwendungsdokuments fordert $o1_1$‘ von der Ausprägungsverwaltung eine Ausprägung des Komponententyps k1 an (5). Die Ausprägungsverwaltung erzeugt eine Ausprägung der Komponente $k1_1$, $k1_1$' genannt und gibt dieses an $o1_1$' zurück. Das Anwendungsdokument wird an $k1_1$‘ übergeben (6). Die Ausprägung $k1_1$' wird mit den im Anwendungsdokument enthaltenen Informationen spezialisiert. (7).

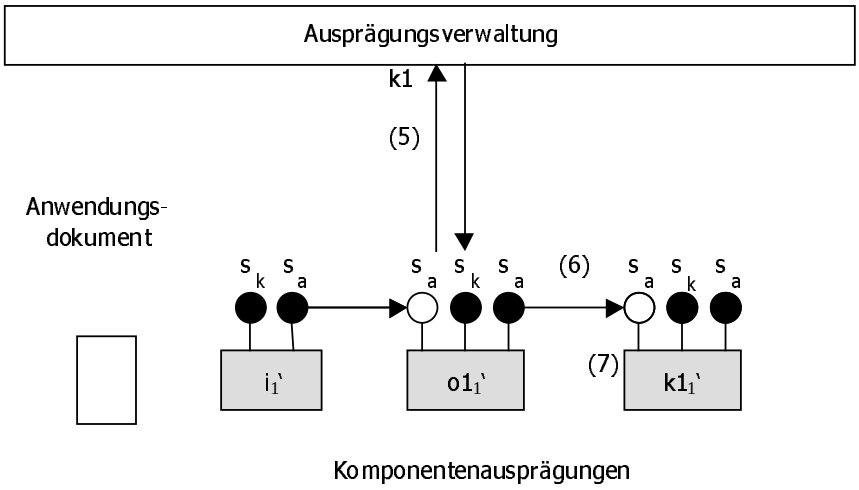


Abbildung 91: Ausführung einer kompositen Anwendung (2)

Den Abschluß bildet die Bildung einer Ausprägung der Endkomponente e_1. Dieser Schritt ist nicht dargestellt. Mit ihr ist die Ausführung der kompositen Anwendung abgeschlossen. Die gesamte Ausprägung der kompositen Anwendung ist in Abbildung 92 wiedergegebn.

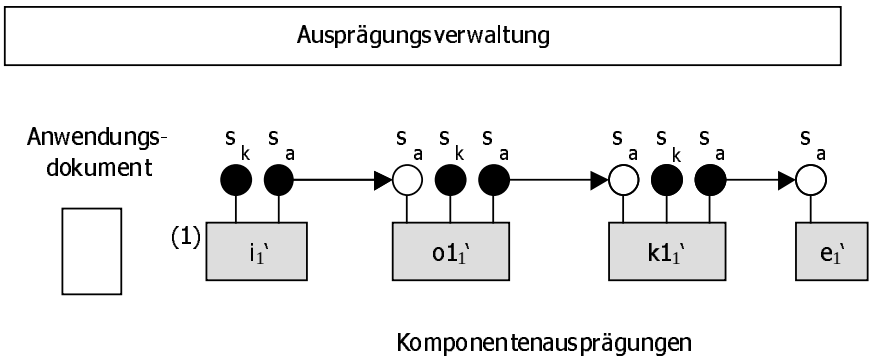


Abbildung 92: Ausführung einer kompositen Anwendung (3)

Der Zeitpunkt ab dem Änderungen bzw. Erweiterungen an kompositen Anwendungen wirksam werden ist wählbar. Die erste Möglichkeit ist die Schaffung eines neuen Anwendungsdokuments. Die bereits gestarteten Ausprägungen der kompositen Anwendung arbeiten mit dem unveränderten Zustand des ursprünglichen Anwendungsdokuments wieder. Die zweite Möglichkeit ist die Erweiterung und Veränderung des Anwendungsdokuments, ohne daß ein neues Anwendungsdokument oder ein neuer Anwendungskontext geschaffen wird. Daher werden auch bereits laufende Ausprägungen der kompositen Anwendung betroffen.

Die Wahl zwischen der Änderung eines bestehenden Anwendungsdokumentes und der Schaffung eines neuen Anwendungsdokumentes kann nur nach Kenntnis der Anwendungssemantik getroffen werden. Daher kann keine allgemeine Regel angegeben werden.

9.3.1.2 Registraturbasierte Spezialisierung

Die zweite neu eingeführte Form der Spezialisierung für komposite Anwendungen ist die registraturbasierte Spezialisierung. Bei ihr werden die Spezialisierungsinformationen in der Registratur abgelegt. Die Spezialisierungsinformationen sind also nicht mehr in einem Anwendungsdokument zentralisiert, sondern auf die Registratureinträge verteilt. Diese können Verknüpfungs- und Parametrisierungsinformationen enthalten. Bei den Verknüpfungsinformationen ist immer auch der Anwendungskontext der Zielkomponente angegeben. Durch die Angabe von Kontextidentifikatoren werden die Spezialisierungsinformationen für unterschiedliche komposite Anwendungen unterschieden. Jedem Komponententyp können unterschiedliche Mengen von Spezialisierungsinformationen zugeordnet werden.

Ein Beispiel für die registraturbasierte Spezialisierung ist in Abbildung 93 dargestellt. Man sieht die Registratur mit den Komponententypen o1 und k1. Ihnen sind die Komponenten $o1_1$ und $k1_1$ zugeordnet. Für beide Komponententypen sind Spezialisierungsinformationen für zwei Kontexte angegeben. Die Spezialisierungsinformationen für Kontext 2 unterscheiden sich im Wert des Parameters Auftragswert. Dieser beträgt im Kontext 1 5.000 Einheiten und im Kontext 2 6.000 Einheiten. Ebenfalls in der Registratur eingetragen sind die Spezialisierungsinformationen für den initialen Komponententyp i. Für den Komponententyp e brauchen keine Spezialisierungsinformationen angegeben zu werden, da es sich um einen finalen Komponententyp handelt.

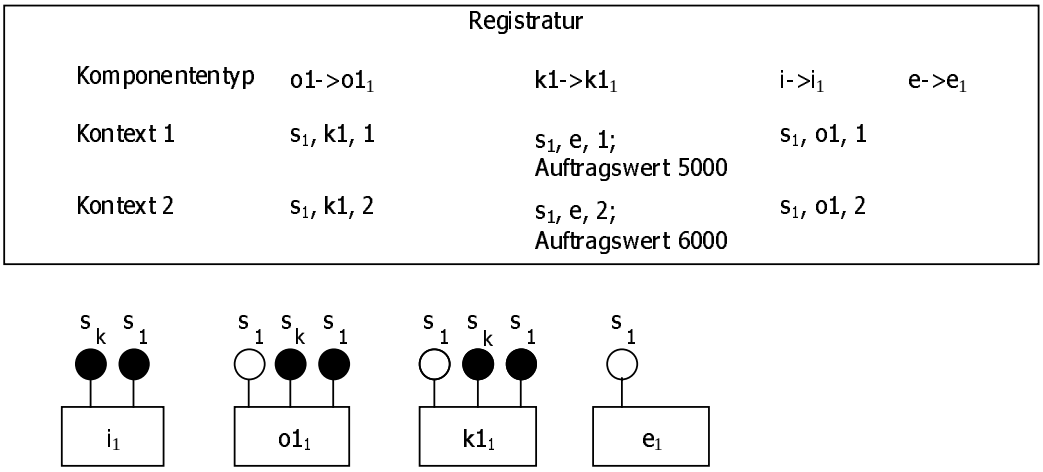


Abbildung 93: Registraturbasierte Spezialisierung

Zur Veranschaulichung wird die Ausführung einer kompositen Anwendung mit Hilfe der registraturbasierten Spezialisierung dargestellt. Ausgangspunkt in Abbildung 94 ist der initiale Komponententyp i_1, dessen Aufgabe es ist, die erste Komponente der kompositen Anwendung zu erzeugen. Die Ausprägung i_1' der Komponente i fordert von der Ausprägungsverwaltung eine Ausprägung des Komponententyps o1 im Kontext 1 an (1). Die Ausprägungsverwaltung fordert daraufhin von der Registratur die erforderlichen Informationen an (2). Nach deren Erhalt (3) erzeugt sie daraufhin eine Ausprägung der Komponente $o1_1$ für den Kontext 1 und gibt dieses an i_1' zurück (4). Daraufhin ruft i_1' die Ausprägung $o1_1$' auf (5).

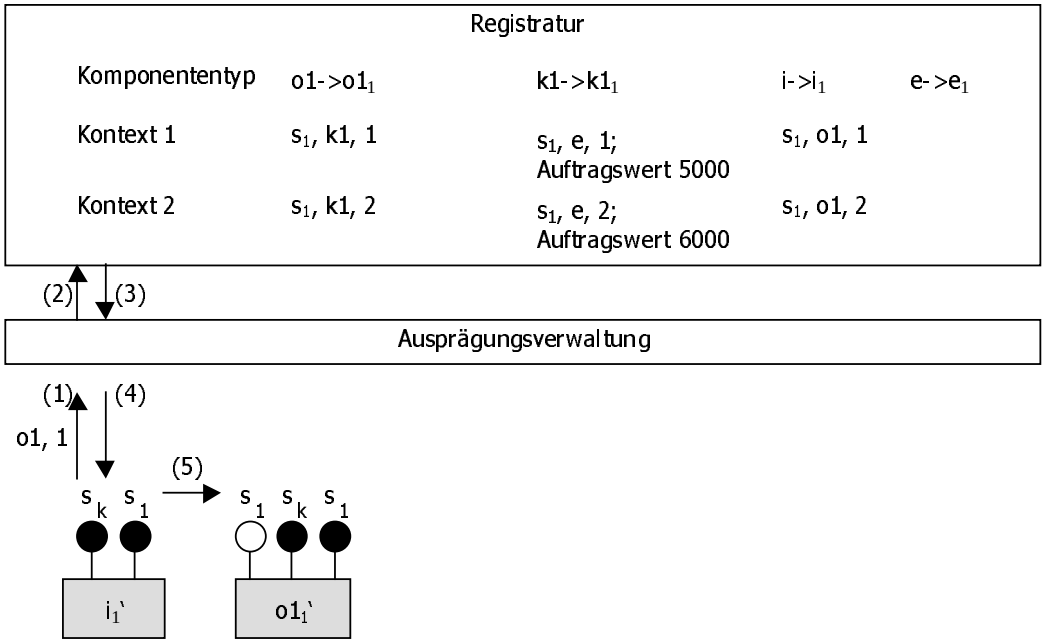


Abbildung 94: Ausführung einer kompositen Anwendung (1)

Durch die in der Registratur enthaltenen Spezialisierungsinformationen ist o1 mit k1 verknüpft. Daher fordert im nächsten Schritt $o1_1$' von der Ausprägungsverwaltung eine Ausprägung des Komponententyps k1 für den Kontext 1 an (6), wie in Abbildung 95 dargestellt. Die Ausprägungsverwaltung erfragt die benötigten Informationen von der Registratur (7) und erzeugt nach deren Erhalt (8) eine Ausprägung der Komponente $k1_1$, $k1_1$' genannt, und gibt dieses an $o1_1$' zurück (9). Anschließend ruft o1' die Ausprägung $k1_1$' auf (10). Die Ausprägung $k1_1$' trägt bereits die Parametrisierungsinformation Auftragswert 5000.

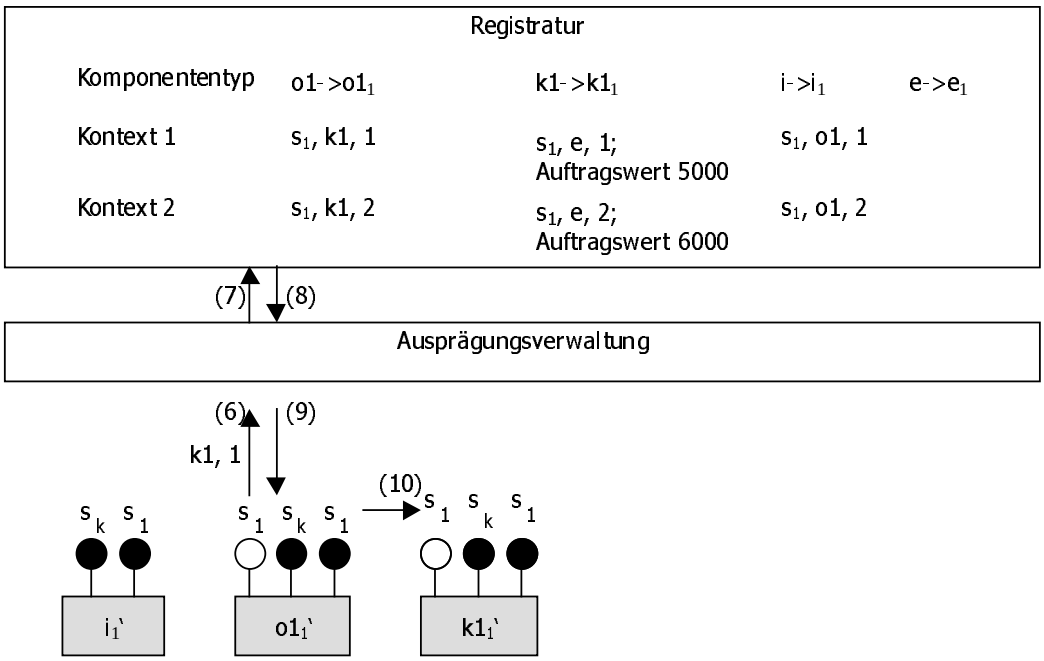


Abbildung 95: Ausführung einer kompositen Anwendung (2)

Den Abschluß bildet die Erzeugung der Komponentenausprägung e_1`. Hierdurch ist die Ausführung der kompositen Anwendung beendet. Dies ist in Abbildung 96 dargestellt, wobei auf die Details der Erzeugung verzichtet wurde.

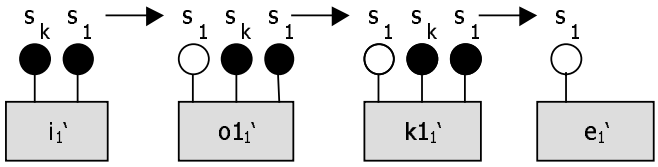
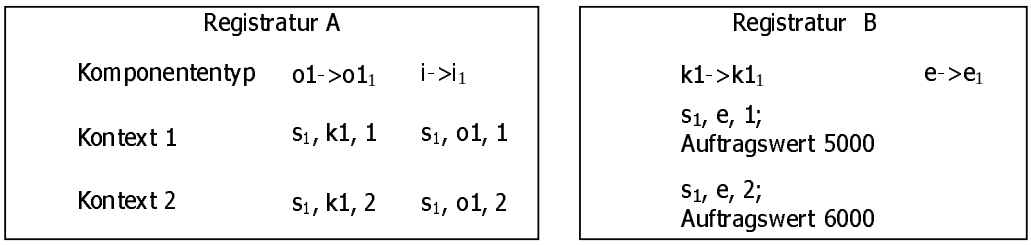


Abbildung 96: Ausführung einer kompositen Anwendung (3)

9.3.1.3 Diskussion der beiden Spezialisierungsmechanismen

Obwohl beide Spezialisierungsmechanismen Vorteile gegenüber der generellen und der individuellen Spezialisierung aufweisen, gibt es entscheidende Unterschiede bezüglich der Eignung für die Unterstützung weitreichender Prozesse. So setzt die anwendungsdokumentbasierte Spezialisierung das Vorhandensein eines zentralen Anwendungsdokumentes voraus. Dieses verletzt jedoch die Forderung nach Autarkie der Prozeßteilnehmer, können sie doch nicht unabhängig voneinander das Anwendungsdokument verändern. Eine Lösung stellt lediglich die Aufteilung des Anwendungsdokumentes in Unterdokumente dar.

Die registraturbasierte Spezialisierung berücksichtigt die Anforderung der Autarkie der Prozeßteilnehmer deutlich besser. So kann jeder Prozeßteilnehmer in seiner Registratur Anpassungen vornehmen, ohne daß er mit anderen Prozeßteilnehmern Rücksprache halten muß. Veranschaulicht ist dies in Abbildung 97. In ihr ist eine mögliche Aufteilung in zwei Registraturen A und B dargestellt.



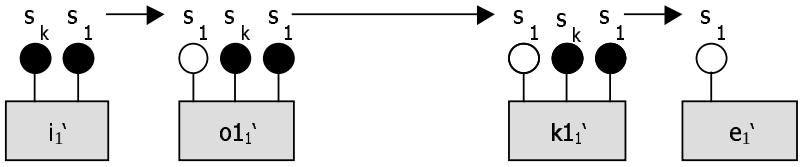


Abbildung 97: Verwendung mehrerer Registraturen

Auf diese Weise kann gewährleistet werden, daß jeder Prozeßteilnehmer eine autarke Zuweisung von Ressourcen durchführen kann, indem er eine eigene und unabhängige Registratur besitzt. In dieser kann er die Ressourcen für die Umsetzung von Diensten frei bestimmen. Darüber hinaus kann er selbst Änderungen von Workflows autark von anderen Teilnehmern durchführen, indem er die Spezialisierungsinformation anpaßt.

9.3.2 Optimierte Ausführung von kompositen Anwendungen

Die bereits bei komponentenorientierten Rahmenwerken beschriebene Optimierung technischer Ressourcen kann auch bei kompositen Anwendungen verwendet werden. Als Ergebnis sind zwei unterschiedliche Ausführungsformen für komposite Anwendungen möglich, die sich in ihrem Ressourcenverbrauch unterscheiden. Diese Ausführungsformen werden als seriell bzw. parallel bezeichnet. Parallel bedeutet, daß die oben dargestellten Ausprägungen gleichzeitig existieren. Dies ist in Abbildung 98 dargestellt. Nachteilig ist der hohe Ressourcenverbrauch.

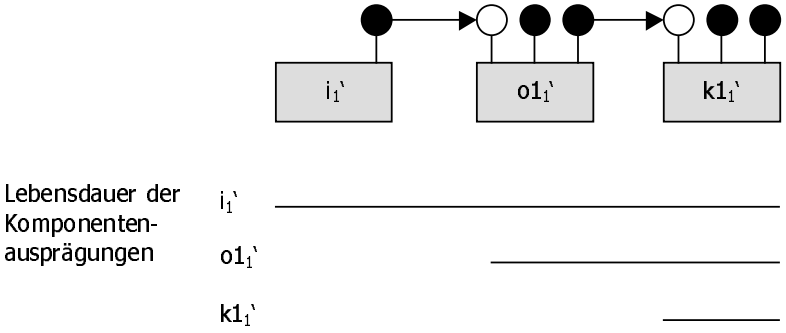


Abbildung 98: Parallele Ausführung einer kompositen Anwendung

Die serielle Ausführung bedeutet, daß eine Komponentenausprägung eine andere Komponentenausprägung aktiviert und sich selbst danach zerstört. Im Beispiel in Abbildung 99 aktiviert i' o1' und wird danach zerstört, dargestellt durch die Schraffierung. Zu diesem Zeitpunkt ist k1‘ noch nicht erzeugt.

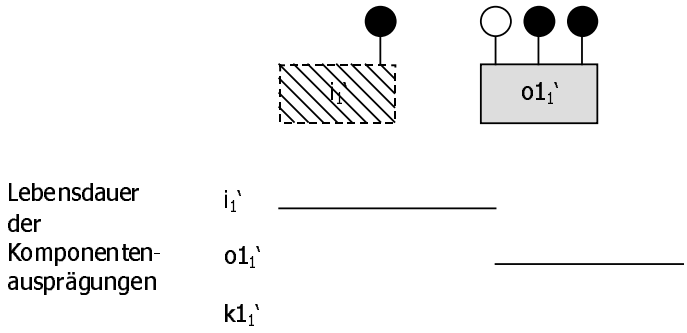


Abbildung 99: Serielle Ausführung einer kompositen Anwendung

Durch die serielle Ausführung kann eine deutliche Reduktion der Anforderungen an technische Ressourcen erreicht werden. So benötigen nicht mehr alle Komponentenausprägungen gleichzeitig Speicher, sondern nutzen nacheinander den Speicher und andere Ressourcen. Dies ist vor allem für Workflows interessant, handelt es sich bei ihnen durch die ihnen inhärente zeitliche Abfolge um in hohem Maße lokale Anwendungen.

9.3.3 Bewertung

Durch Veränderung des Anwendungsdokumentes bzw. der registraturbasierten Spezialisierungsinformationen ist es möglich, Anwendungen um zusätzliche Dienste zu erweitern bzw. die Verbindung der Dienste untereinander zu verändern. Die Anforderungen der Anwendungserweiterung und -konfiguration werden daher von kompositen Anwendungen erfüllt.

Zur Veranschaulichung wird die oben vorgestellte komposite Anwendung um die Komponententypen o2 und o3 erweitert. Vom Komponententyp k1 aus sollen Verknüpfungen zu diesen Komponententypen hergestellt werden. Beide Komponententypen sollen mit dem Komponententyp k1 verknüpft werden. Die Verknüpfung von k1 mit e soll dafür gelöst werden. Zusätzlich sollen die neu hinzugekommenen Komponententypen o2 und o3 mit e verknüpft werden.

Zur Umsetzung werden die Spezialisierungsinformationen von Komponententyp k1 so erweitert, daß sie Verknüpfungen mit den Komponententypen o2 und o3 enthalten. Die Erweiterungen sind in Abbildung 100 herausgehoben. Im Gegenzug wird die Verknüpfung mit dem Komponententyp e gelöscht. Die Komponententypen o2 und o3 sollen alternativ aktiviert werden. Die Entscheidung hierüber trifft der Komponententyp k1 in Abhängigkeit von einem Entscheidungsparameter. Dieser Entscheidungsparameter wird ebenfalls in den Spezialisierungsinformationen des Komponententyps k1 abgelegt. Der Name des Entscheidungsparameters ist Auftragswert, der Entscheidungswert ist 5000.

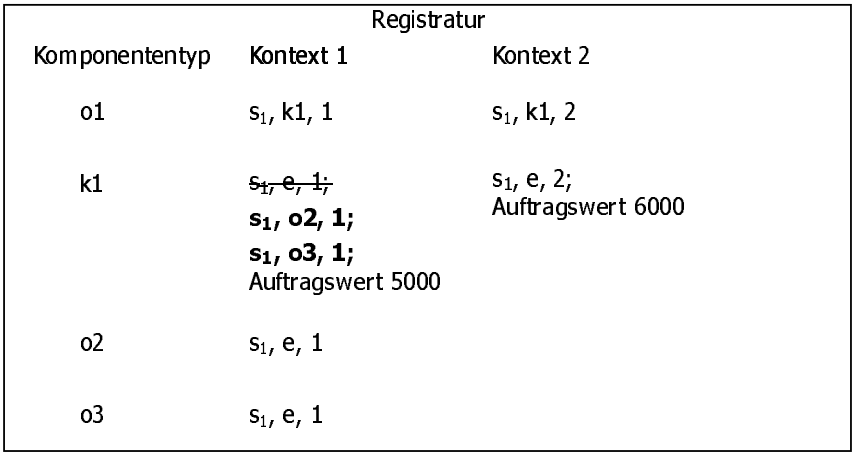


Abbildung 100: Anwendungserweiterbarkeit und -konfigurierbarkeit

9.4 Zusammenfassung

In diesem Kapitel wurde mit Komponentensystemen eine Software-Architektur eingeführt, die die in den Kapiteln 6 bis 8 identifizierten nicht-funktionalen Anforderungen erfüllt. Die Erfüllung einzelner Anforderungen durch die Bestandteile von Komponentensystemen ist in Tabelle 3 dargestellt. In den Kreuzungspunkten von Zeilen und Spalten ist jeweils das Konzept oder die Funktion angegeben, die die jeweilige Anforderung erfüllt.

	Ressourcen	Dienste	Anwendungen
Erweiterbarkeit	Hinzufügen von Komponenten	Hinzufügen von Komponenten-typen	Erweiterung einer kompositen Anwendung
Konfigurierbarkeit	Veränderung der Zuordnung Komponententyp - Komponente	Spezialisierungs-mechanismen	Anpassung einer kompositen Anwendung

Tabelle 3: Erfüllung der nicht-funktionalen Anforderungen

Ausgangspunkt war die Entwicklung eines neuen Begriffsapparates für Komponenten und komponentenorientierte Rahmenwerke, der es ermöglicht zu klären, welche der identifizierten nicht-funktionalen Anforderungen von Komponenten und komponentenorientierten Rahmenwerken erfüllt werden. Dabei stellte sich heraus, daß Komponenten und komponentenorientierte Rahmenwerke allein nicht alle Anforderungen erfüllen.

Daher wurde das Konzept der kompositen Anwendung entwickelt. Komposite Anwendungen wurden erstmalig in [Schm97a] beschrieben und hier in einer erweiterten Form dargestellt. Komposite Anwendungen erfüllen die Anforderungen der unabhängigen Anwendungserweiterbarkeit und –konfigurierbarkeit. Eine komposite Anwendung stellt eine Laufzeitabstraktion dar, indem sie vom Verhalten und der Zusammenarbeit von Komponentenausprägungen abstrahiert. In kompositen Anwendungen wird von konkreten Komponenten abstrahiert und es werden nur Komponententypen referenziert. Durch Veränderung der Verknüpfungen zwischen den Komponententypen können Anwendungskonfigurationen und –erweiterungen leicht umgesetzt werden. Komposite Anwendungen sind zudem verträglich zu komponentenorientierten Rahmenwerken und Komponenten gestaltet, so daß die Anforderungen der Diensteerweiterbarkeit und –konfigurierbarkeit sowie der Ressourcenerweiterbarkeit und –konfigurierbarkeit er-

füllt werden. Darüber hinaus bieten sie eine deutlich verbesserte Ressourcennutzung, wie Abschnitt 9.3.2 gezeigt hat.

Für komposite Anwendungen wurden zwei neue Spezialisierungsmechanismen entwickelt, die anwendungsdokumentbasierte und die registraturbasierte Spezialisierung. Die registraturbasierte Spezialisierung ist besonders gut für die Zwecke dieser Arbeit geeignet, sieht sie doch die Verteilung der Spezialisierungsinformationen auf die Registratureinträge der Komponententypen vor. Hierdurch können die Spezialisierungsinformationen einer kompositen Anwendung auf mehrere Registraturen verteilt werden. Diese können verteilt sind, also sich beispielsweise bei verschiedenen Teilnehmern eines Prozesses befinden. Daher kommt die registraturbasierte Spezialisierung in besonderem Maße der Anforderung nach Autarkie der Prozeßteilnehmer entgegen.

10 Aspektorientierte Komponentensysteme als Realisierung

In den Kapiteln 6, 7 und 8 wurden Lösungskonzepte für die in Kapitel 5 identifizierten Problembereiche entwickelt. Aus diesen Lösungskonzepten wurden nicht-funktionale Anforderungen entwickelt, die von einer Software-Architektur erfüllt werden müssen, damit sie zur Umsetzung der Lösungskonzepte prinzipiell in der Lage ist. Es sind dies die Anforderungen der unabhängigen Dienste- und Ressourcenerweiterbarkeit, der Dienste- und Ressourcenkonfigurierbarkeit, sowie der unabhängigen Anwendungserweiterbarkeit und –konfigurierbarkeit. In Kapitel 9 wurde mit Komponentensystemen eine Software-Architektur konzipiert, die diese Anforderungen erfüllt und dadurch als Grundlage für die Realisierung der Lösungskonzepte tauglich ist.

In diesem Kapitel kann daher die Realisierung der Lösungskonzepte durchgeführt werden. Die sich dabei ergebende Gesamtarchitektur wird als aspektorientiertes Komponentensystem bezeichnet. Ein aspektorientiertes Komponentensystem ist ein Komponentensystem mit Aspekten und Aspektelementen als Granularitätskriterium. Dieses Granularitätskriterium wird durch die Lösungskonzepte bestimmt. Die Realisierung der Lösungskonzepte erfolgt getrennt für die aspektelementorientierte Schemarepräsentation und das Workflow-Modell-Wörterbuch.

Erste Aufgabe bei der Realisierung der aspektorientierten Schemarepräsentation ist es, eine Analyse durchzuführen, welche Bestandteile der aspektelementorientierten Schemarepräsentation durch welche Bestandteile von Komponentensystemen realisiert werden können. Es gilt also in beiden korrespondierende Bestandteile zu identifizieren. Basierend auf den Ergebnissen der Analyse wird dann die konkrete Abbildung der korrespondierenden Bestandteile durchgeführt. Dazu werden Abbildungsregeln definiert, die die Realisierung von Bestandteilen der aspektelementorientierten Schemarepräsentation durch als korrespondierend erkannte Bestandteile beschreiben. Die Herausforderung liegt darin, die Abbildung so durchzuführen, daß Topologie- und Parameteränderungen als auch Schemaerweiterungen- und –reduktionen vorgenommen werden können.

Die Realisierung des Workflow-Modell-Wörterbuchs könnte zunächst unabhängig von Komponentensystemen erfolgen. Dabei würde es jedoch zu redundanten Informationen im Workflow-Modell-Wörterbuch und der Registratur des Komponentensystems kommen. Die Registratur von Komponentensystemen enthält explizit oder implizit eine Reihe von Informationen, die auch im Workflow-Modell-Wörterbuch anzutreffen sind. Aus diesem Grund liegt die Herausforderung bei der Realisierung des Workflow-Modell-Wörterbuchs in der Bestimmung der Informationen, die bereits in der Registratur vorhanden sind, und deren Integration in das

Workflow-Modell-Wörterbuch. Hierfür wird eine Analyse durchgeführt, welche Informationen in der Registratur für das Workflow-Modell-Wörterbuch nutzbar sind. Als Ergebnis entsteht ein Integrationsmodell für die Integration registraturbasierter Informationen in das Workflow-Modell-Wörterbuch.

10.1 Realisierung der aspektelementorientierten Schemarepräsentation

Zur Realisierung der aspektelementorientierten Schemarepräsentation wird zunächst die Identifizierung korrespondierender Bestandteile in der aspektelementorientierten Schemarepräsentation und Komponentensystemen durchgeführt. Der zweite Schritt ist die Definition konkreter Abbildungsregeln.

10.1.1 Identifizierung korrespondierender Bestandteile

Die Analyse, welche Bestandteile der aspektelementorientierten Schemarepräsentation mit Bestandteilen kompositer Anwendungen korrespondieren, erfolgt zweistufig. Zunächst wird analysiert, wie eine aspektelementorientierte Schemarepräsentation auf das Konzept der Anwendung, nach der Definition von Kapitel 2, abgebildet werden kann. In einem zweiten Schritt geschieht dann die Abbildung auf die Bestandteile kompositer Anwendungen.

Die aspektelementorientierte Schemarepräsentation besteht aus Repräsentationselementen, Verbindungspunkten und Verbindern. Die Repräsentationselemente der aspektelementorientierten Schemarepräsentation stehen für Elemente physischer Aspekte. Greift man die Struktur des Zwei-Ebenen-Workflow-Metamodells auf, dann sieht man, daß sich hinter Elementen physischer Aspekte eine Menge von Diensten verbirgt, die wiederum durch eine oder mehrere Ressourcen bereitgestellt werden. Aus diesem Grund entsprechen also die Repräsentationselemente einer Menge von Diensten, die in ihrer Gesamtheit ein Element eines Aspektes unterstützen. Die Zusatzinformationen der Repräsentationselemente ermöglichen die Anpassung von Entscheidungswerten. Sie entsprechen der Parametrisierung von Diensten in einer Anwendung. Die Verbinder der aspektelementorientierten Schemarepräsentation verbinden Repräsentationselemente an ihren Verbindungspunkten. Sie können daher der Verknüpfung von Diensten gleichgesetzt werden. Dementsprechend können dann die Verbindungspunkte der Repräsentationselemente Diensten gleichgesetzt werden. Veranschaulicht sind diese Zusammenhänge in Abbildung 101.

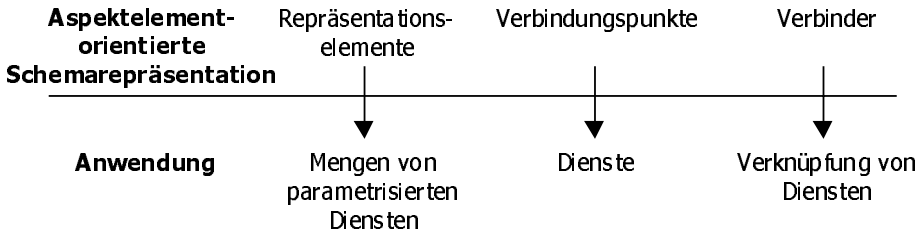


Abbildung 101: Schemarepräsentation / Anwendung

Der nächste Schritt ist die Identifikation korrespondierender Bestandteile zwischen der aspektelementorientierten Schemarepräsentation und der kompositen Anwendung. Die Ergebnisse sind in Abbildung 102 veranschaulicht. Im ersten Analyseschritt wurde festgestellt, daß Repräsentationselemente für eine Menge von parametrisierten Diensten stehen. In kompositen Anwendungen findet sich eine Entsprechung in Form von Komponententypen mit ihren Spezialisierungsinformationen. Ein Komponententyp steht für eine Menge von Diensten, die

jeweils über Schnittstellen erbracht werden. Durch Angabe von Parametrisierungsinformationen können die Dienste in ihren Eigenschaften angepaßt werden. Die Repräsentationselemente der aspektelementorientierten Schemarepräsentation werden daher in der kompositen Anwendung durch einen Komponententyp und dementsprechende Parametrisierungsinformationen realisiert. Die Verbindungspunkte der aspektelementorientierten Schemarepräsentation sind Ansatzpunkte für Verbindungen zwischen Repräsentationselementen. Es liegt daher nahe, Schnittstellen von Komponenten als korrespondierendes Konzept in kompositen Anwendungen zu identifizieren. Dementsprechend können die Verbinder der aspektelementorientierten Schemarepräsentation mit den Verknüpfungsinformationen einer kompositen Anwendung gleichgesetzt werden.

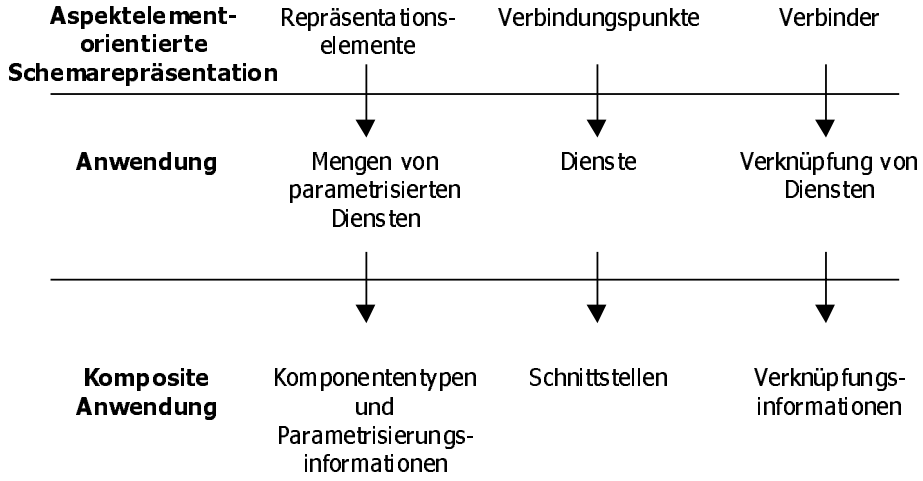


Abbildung 102: Schemarepräsentation / komposite Anwendung

10.1.2 Abbildungsregeln

Nachdem festgestellt worden ist, welche Bestandteile der aspektelementorientierten Schemarepräsentation mit welchen Bestandteilen kompositer Anwendungen gleichzusetzen sind, werden Abbildungsregeln für die Bestandteile der aspektelementorientierten Schemarepräsentation festgelegt. Dabei gilt es genau festzulegen, wie Repräsentationselemente, Verbindungspunkte und Verbinder auf Komponententypen, Schnittstellen sowie Parametrisierungs- und Verknüpfungsinformationen abzubilden sind.

10.1.2.1 Repräsentationselemente und Verbindungspunkte

Für die Abbildung von Repräsentationselementen ist zunächst festzulegen, daß ein Repräsentationselement immer auf einen entsprechenden Komponententyp abzubilden ist. Dieser Komponententyp muß dem Repräsentationselement in seiner Aspektzugehörigkeit entsprechen. Außerdem muß der Komponententyp dem vom Repräsentationselement repräsentierten Aspektelement entsprechen.

Die beiden Typen von Verbindungspunkten in der aspektelementorientierten Schemarepräsentation, temporale und nicht-temporale Verbindungspunkte, werden mit zwei grundsätzlich unterschiedlichen Typen von Schnittstellen ausgestattet. Temporale Verbindungspunkte werden in der kompositen Anwendung durch eine ausgezeichnete Schnittstelle s_z umgesetzt. Die Schnittstelle s_z dient zur Wiedergabe der zeitlichen Abfolge, die durch temporale Verbinder zwischen zwei temporalen Verbindungspunkten definiert wird. In allen Komponententypen, die über temporale Verbinder verknüpft sind, taucht daher die Schnittstelle s_z auf. Eine Ausnahme bilden jeweils der erste oder der letzte Komponententyp einer kompositen Anwendung.

Die nicht-temporalen Verbindungspunkte werden durch individuelle Schnittstellen wiedergegeben. Da sie individuell gestaltete Dienstnehmer-/Dienstgeberbeziehungen unterstützen, wird jeder nicht-temporale Verbindungspunkt durch eine eigene Schnittstelle unterstützt. Diese Beziehung spiegelt sich als ausgehende Schnittstelle beim dienstnehmenden Komponententyp und als eingehende Schnittstelle beim dienstgebenden Komponententyp wider.

Auf der Basis des Vorhandenseins der Schnittstelle s_z und von individuellen Schnittstellen können Komponentenarten identifiziert werden. Dazu wird untersucht, welche Kombinationen der Schnittstelle s_z und von individuellen Schnittstellen auftreten. Ein zusätzliches Unterscheidungskriterium ist, ob die Schnittstellen in eingehender oder ausgehender Form oder in beiden Formen auftritt. Eine Übersicht gibt Tabelle 4. Sie dient Gemeinsamkeiten von Komponenten auf der Basis ihrer Schnittstellen zu identifizieren. Dazu teilt sie die Komponenten in unterschiedliche Arten ein, die sich durch das Vorhandensein bzw. das Fehlen bestimmter Schnittstellen definieren. Das Vorhandensein einer Schnittstelle ist mit einem Pluszeichen dargestellt. Das Fehlen einer Schnittstelle ist mit einem Minuszeichen wiedergegeben. Ist das Vorhandensein der Schnittstelle freigestellt, so ist ein Kreiszeichen eingetragen.

Komponenten mit nur einer ausgehenden Schnittstelle s_z werden als initiale Komponenten bezeichnet. Von ihnen geht die Ausführung einer kompositen Anwendung aus. Entsprechend gibt es auch finale Komponenten. Finale Komponenten besitzen nur eine eingehende Schnittstelle s_z und beenden die Ausführung einer kompositen Anwendung.

Temporale Komponenten besitzen die Schnittstelle s_z in ein- und ausgehender Form. Individuelle Schnittstellen können bei ihnen in ausgehender Form auftreten, müssen aber nicht.

Komponenten, die nur individuelle Schnittstellen in ein- und ausgehender Form oder nur eingehender Form besitzen, werden als Dienstkomponenten bezeichnet. Darüber hinaus gibt es noch weitere Fälle für die Kombination der Schnittstellen, die allerdings wenig sinnvoll sind und nur der Vollständigkeit halber erwähnt werden. Den Anfang macht der Fall, daß keine Schnittstelle vorhanden ist. Eine derartige „Komponente“, wenn man vor ihr überhaupt noch als solcher sprechen kann, ist ohne Funktion.

	s_z eingehend	s_z ausgehend	individuelle Schnittstelle eingehend	individuelle Schnittstelle ausgehend
Initiale Komponente	-	+	-	-
Finale Komponente	+	-	-	-
Temporale Komponenten	-	+	-	+
	+	-	-	+
	+	+	-	o
Komponenten mit impliziten temporalen Abhängigkeiten	+	-	+	o
	-	+	+	o
	+	+	+	-
	+	+	+	+
Dienstkomponenten	-	-	+	+
	-	-	+	-

Nutzlose Kombinationen	-	-	-	-

Tabelle 4: Komponentenarten

10.1.2.2 Verbinder

Auch für die Verbinder sind zwei Abbildungsregeln zu definieren. Temporale Verbinder werden als Verknüpfungsinformationen für die Schnittstelle s_z wiedergegeben. Hierzu werden die Nachfolger eines Komponententyps in der kompositen Anwendung in die Verknüpfungsinformationen der Schnittstelle s_z eingetragen. Die Schnittstelle s_z ist also bei diesem Komponententyp eine ausgehende Schnittstelle, bei den Nachfolgern eine eingehende Schnittstelle. Sind mehrere Nachfolger vorhanden, so muß eine Auswahlfunktion entscheiden, welche der Nachfolger aktiviert werden. Hierzu können zusätzliche Parametrisierungsinformationen im Anwendungsdokument angegeben werden.

Die nicht-temporalen Verbinder der aspektelementorientierten Schemarepräsentation, die Dienstnehmer-/Dienstgeberbeziehungen wiedergeben, werden ebenfalls in die Verknüpfungsinformationen des dienstnehmenden Komponententyps eingetragen. Sie schlagen sich in Verknüpfungsinformationen für Schnittstellen außer s_z nieder. Beim Dienstnehmer ist die individuelle Schnittstelle eine ausgehende Schnittstelle, beim Dienstgeber eine eingehende Schnittstelle.

10.1.3 Realisierungsschritte

Die Realisierung wird in drei Schritten durchgeführt. Zunächst werden unter Anwendung der entwickelten Abbildungsregeln Komponententypen spezifiziert. Im zweiten Schritt werden die Spezialisierungsinformationen, bestehend aus Parametrisierungs- und Verknüpfungsinformationen, erstellt. Der dritte Schritt ist die Anpassung der Spezialisierungsinformationen an die Bedürfnisse der Anwendungsdokument- bzw. registraturbasierten Spezialisierung. Zur Veranschaulichung wird dabei das bereits aus den vorangegangenen Kapiteln bekannte Beispiel umgesetzt, das in Abbildung 103 dargestellt ist. Zur Bezugnahme im Text sind die temporalen Verbinder mit den Buchstaben a bis f und die nicht-temporalen Verbinder mit den Ziffern 1 bis 4 identifiziert.

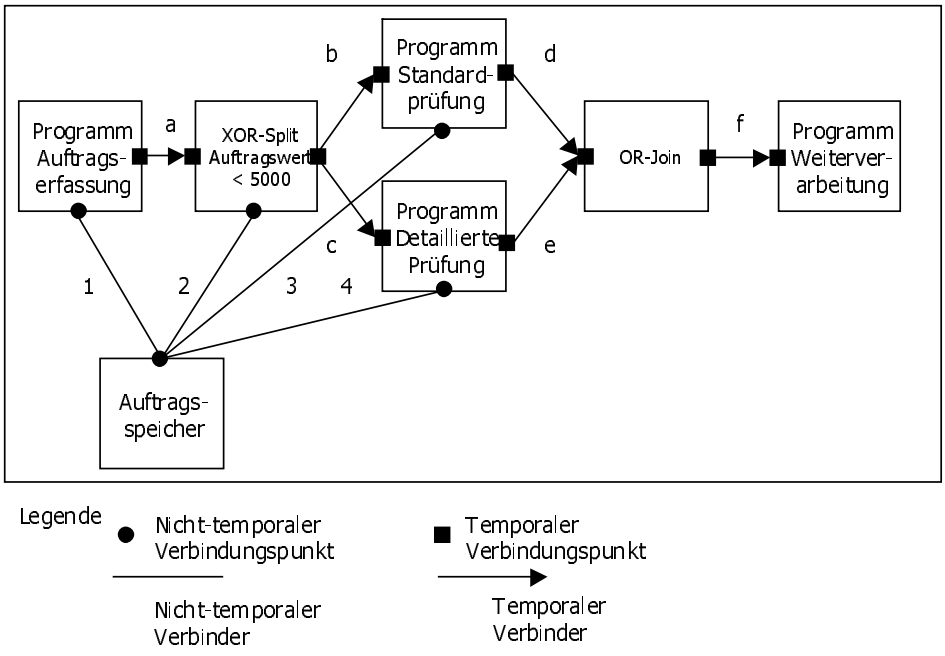


Abbildung 103: Aspektelementorientierte Schemarepräsentation des Beispiels

10.1.3.1 Spezifikation der Komponententypen

Die Repräsentationselemente der dynamischen Schemarepräsentation und der dafür vorgesehenen Komponenten der kompositen Anwendung sind in Tabelle 5 gegenübergestellt. Es wird angenommen, daß die Repräsentationselemente "Programm Auftragserfassung", "Programm Standardprüfung", "Programm Detaillierte Prüfung" und „Programm Weiterverarbeitung“ durch die Komponententypen oa, os, od und ow unterstützt werden. Die Repräsentationselemente der dynamischen Schemarepräsentation, die Elemente des Kontrollaspekts repräsentieren, werden durch die Komponententypen ks und kj unterstützt. Der Komponententyp ks ist ein Gabelungselement. Das Verknüpfungselement kj führt die Zusammenführung, also einen OR-Join durch. Mit dem Komponententyp da wird das Repräsentationselement des Datenaspekts "Auftrag" umgesetzt. Die weitere Verarbeitung wird dann vom Komponententyp ow durchgeführt.

Repräsentationselemente	**Komponententypen**
Programm Auftragserfassung	oa
Programm Standardprüfung	os
Programm Detaillierte Prüfung	od
Gabelungselement XOR-Split	ks
Verknüpfungselement OR-Join	kj
Auftragsspeicher	da
Programm Weiterverarbeitung	ow

Tabelle 5: Repräsentationselemente und Komponententypen

Im folgenden werden die oben aufgezählten Komponententypen näher beschrieben. In Abbildung 104 sind die Komponententypen und ihre Zugehörigkeit zu den einzelnen Aspekten dargestellt. Der initiale Komponententyp i und der Endkomponententyp e werden im folgenden nur dargestellt, wenn es notwendig ist. Auch die Schnittstelle s_k, die zur Anforderung von Komponenten dient, ist nicht dargestellt.

Die Komponententypen oa, od, os und ow sind dem Operationsaspekt zugehörig. Sie enthalten die Schnittstelle s_z mit der sie aktiviert werden, bzw. ihren Nachfolger aktivieren. Die Schnittstelle s_1 ist eine ausgehende Schnittstelle, die zum Ansprechen der benötigten Daten in dem Komponententyp da, also dem Auftragsspeicher, dient.

Komponenten des Typs ks und kj sind dem Kontrollaspekt zugehörig. Der Komponententyp ks besitzt eine eingehende Schnittstelle s_z zu ihrer Aktivierung und zwei ausgehende Schnittstellen. Die Schnittstelle s_z dient zu ihrer Aktivierung und die Schnittstelle s_1 zur Abfrage des als Entscheidungskriterium dienenden Datums. Mit Hilfe von Parametrisierungsinformationen wird Name und Wert des als Entscheidungskriteriums dienenden Wertes angegeben. Die in den Verknüpfungsinformationen von s_z angegebenen Schnittstellen werden in Abhängigkeit vom Ergebnis des Tests aktiviert. Der Komponententyp kj besitzt eine eingehende Schnittstellen s_z und eine ausgehende Schnittstelle s_z.

Der einzige Komponententyp, der dem Datenaspekt zugehörig ist, ist der Komponententyp da. Er verfügt nur über eine eingehende Schnittstelle s_1. Der Anschaulichkeit halber wird ange-

nommen, daß durch sie direkt der Zugriff auf Daten erfolgen kann, ohne daß weitere Komponenten verwendet werden müssen. Für ihn brauchen daher auch keine Spezialisierungsinformationen gebildet zu werden.

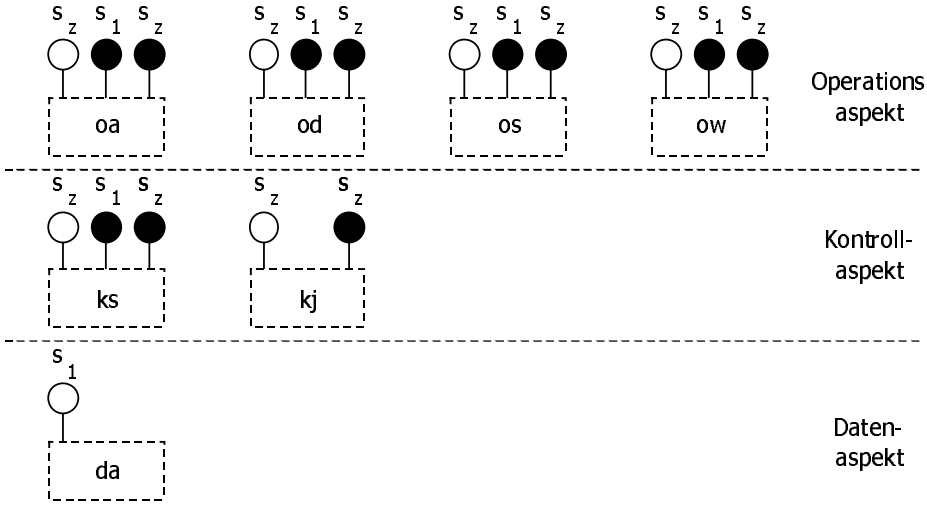


Abbildung 104: Komponententypen

10.1.3.2 Erzeugung von Spezialisierungsinformationen

Nachdem die benötigten Komponententypen beschrieben worden sind, können die Spezialisierungsinformationen schrittweise entwickelt werden. Dies geschieht, indem Zusatzinformationen der Repräsentationselemente zu Parametrisierungsinformationen und Verbinder der aspektelementorientierten Schemarepräsentation zu Verknüpfungselementen umgesetzt werden.

Ausgangspunkt ist das Repräsentationselement „Programm Auftragserfassung“, das durch den Komponententyp oa umgesetzt wird. Es steht über den temporalen Verbinder a mit dem Repräsentationselement „Gabelungselement XOR-Split“ in Verbindung. Über den nicht-temporalen Verbinder 1 ist es außerdem mit dem Repräsentationselement Auftragsspeicher verbunden. Die Verbinder werden durch die Spezialisierungsinformationen für den Komponententyp oa, dargestellt in Tabelle 6, umgesetzt. Sie enthalten Verknüpfungen mit den Komponententypen ks und da. So ist der Komponententyp oa mit dem Komponententyp ks über die Schnittstelle s_z verbunden. Mit dem Komponententyp da ist oa über die Schnittstelle s_1 verbunden.

Komponententyp	Spezialisierungsinformationen
oa	s_z, ks; s_1, da

Tabelle 6: Spezialisierungsinformationen (1)

Im nächsten Schritt wird der Verbinder des Repräsentationselementes „Gabelungselement XOR-Split“ umgesetzt. Es gilt also die temporalen Verbinder b und c sowie den nicht-temporalen Verbinder 2 umzusetzen. Der Komponententyp ks muß daher mit den Komponententypen os und od über die Schnittstelle s_z verbunden werden. Die Spezialisierungsinformationen von ks enthalten zusätzlich die Information, daß der Parameter Auftrag auf den Wert 5000 geprüft werden soll. Außerdem ist eine Verknüpfung zum Komponententyp da notwendig, um den nicht-temporalen Verbinder 2 umzusetzen. Dies geschieht über die Schnittstelle s_1. Die Spezialisierungsinformationen für den Komponententyp ks werden in Tabelle 6 aufgenommen, das Ergebnis ist in Tabelle 7 dargestellt.

Komponententyp	Spezialisierungsinformationen
oa	$s_{z,}$ ks ; $s_{1,}$ da
ks	$s_{z,}$ os ; $s_{z,}$ od ; $s_{1,}$ da; Auftragswert, 5000

Tabelle 7: Spezialisierungsinformationen (2)

Die Repräsentationselemente „Programm Standardprüfung“ und „Programm Detaillierte Prüfung“ sind über die temporalen Verbinder d und e mit dem Repräsentationselement „Verknüpfungselement OR-Join“ verbunden. Die Komponententypen os und od, die die Repräsentationselemente umsetzen, müssen daher mit dem Komponententyp kj verknüpft werden. Dies geschieht jeweils über die Schnittstelle s_z. Außerdem müssen bei beiden Repräsentationselementen die nicht-temporalen Verbinder 3 und 4 zum Repräsentationselement Auftragsspeicher umgesetzt werden. Dies geschieht über eine Verknüpfung der Komponenten os und od mit dem Komponententyp da. Die Erweiterung ist in Tabelle 8 dargestellt.

Komponententyp	Spezialisierungsinformationen
oa	$s_{z,}$ ks ; $s_{1,}$ da
ks	$s_{z,}$ os ; $s_{z,}$ od ; $s_{1,}$ da; Auftragswert, 5000
os	$s_{z,}$ kj ; $s_{1,}$ da
od	$s_{z,}$ kj ; $s_{1,}$ da

Tabelle 8: Spezialisierungsinformationen (3)

Schließlich muß der Verbinder f des Repräsentationselementes „Verknüpfungselement OR-Join“ mit dem Repräsentationselement „Programm Weiterverarbeitung“ umgesetzt werden. Hierzu ist der temporale Verbinder f umzusetzen. Er wird in dementsprechende Verknüpfungsinformationen für den Komponententyp kj umgesetzt. Die Schnittstelle hierfür ist wiederum $s_{z.}$ Der Komponententyp ow ist seinerseits mit dem Endkomponententyp e verbunden. Das Ergebnis sieht man in Tabelle 9.

Komponente	Spezialisierungsinformationen
oa	$s_{z,}$ ks; $s_{1,}$ da
ks	$s_{z,}$ os; $s_{z,}$ od; $s_{1,}$ da; Auftragswert, 5000
os	$s_{z,}$ kj; $s_{1,}$ da
od	$s_{z,}$ kj; $s_{1,}$ da
kj	$s_{z,}$ ow
ow	$s_{z,}$ e

Tabelle 9: Spezialisierungsinformationen (4)

10.1.3.3 Anpassung der Spezialisierungsinformationen

Den Abschluß bildet die Anpassung der Spezialisierungsinformationen für die Bedürfnisse der anwendungsdokument- und registraturbasierten Spezialisierung. Bei der anwendungsdokumentbasierten Spezialisierung gestaltet sich dieser Schritt sehr einfach. Die er-

mittelten Spezialisierungsinformationen können einfach in ein Anwendungsdokument übertragen werden. Durch die Möglichkeit der Mehrfachnennung von Komponententypen und die Angabe einer Reihenfolge werden unterschiedliche Spezialisierungsinformationen für den gleichen Komponententyp auch dann sicher getrennt, wenn dieser mehrfach verwendet wird.

Bei der Erzeugung von Spezialisierungsinformationen für die registraturbasierte Spezialisierung taucht jedoch ein Problem auf. Wird eine Komponente mehrfach und mit unterschiedlichen Spezialisierungsinformationen verwendet, werden unterschiedliche Kontextidentifikatoren benötigt. Dieses Problem wird als Kontextproblem bei registraturbasierter Spezialisierung bezeichnet.

Zur Veranschaulichung der Problematik sind in Abbildung 105 die Spezialisierungsinformationen aus Tabelle 9 für die registraturbasierte Spezialisierung aufbereitet worden. Dazu sind sie um den Kontextidentifikator 1 ergänzt worden und in die Registratur aufgenommen worden. Zu jedem Komponententyp sind entsprechende Komponenten eingetragen. Es sind dies oa_1 für den Komponententyp oa, ks_1 für den Komponententyp ks usw. Die Spezialisierungsinformationen für weitere Anwendungskontexte sind nicht dargestellt, um die Übersichtlichkeit nicht zu stören. Aus Platzgründen ist außerdem der Registratureintrag für den Komponententyp da nicht dargestellt. Da dieser Komponententyp keine Spezialisierungsinformationen benötigt, sind die im folgenden angestellten Betrachtungen für ihn nicht relevant.

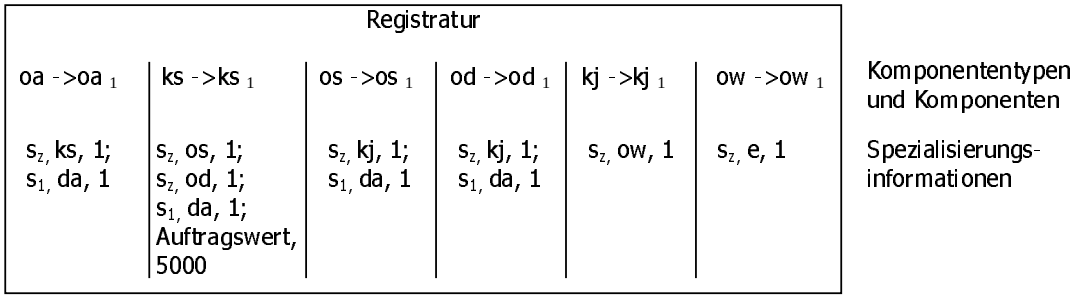


Abbildung 105: Registratur mit Spezialisierungsinformationen

Das Kontextproblem taucht auf, wenn ein Komponententyp in einer kompositen Anwendung, also in einem Anwendungskontext, zweimal verwendet wird. Als Beispiel wird dazu die zweifache Verwendung des Komponententyps ks betrachtet. Es wird angenommen, daß für die zweite Verwendung des Komponententyps ks eine Verknüpfung mit Komponenten vom Typ ob und ow durchgeführt werden soll. Daher ergeben sich entsprechend veränderte Verknüpfungsinformationen. Ebenfalls wird angenommen, daß der Entscheidungsparameter nun Embargo lautet und der Entscheidungswert „ok“ beträgt. Trägt man diese Informationen in die Registratur ein, so kommt es zu der in Abbildung 106 dargestellten Situation, daß für den Komponententyp ks zwei Mengen von Spezialisierungsinformationen zur Verfügung stehen, ohne daß klar ist, welche verwendet werden soll.

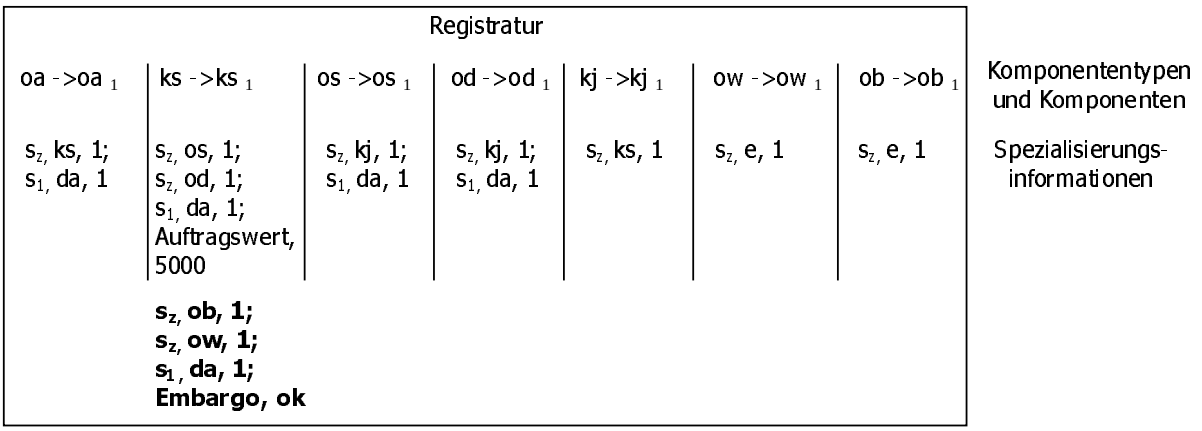


Abbildung 106: Kontextproblem bei registraturbasierter Spezialisierung

Um diese Vieldeutigkeit zu verhindern, muß der Kontextidentifikator gewechselt werden. Dies ist in Abbildung 107 dargestellt. So ist für die zusätzlichen Spezialisierungsinformationen des Komponententyps ks der Kontextidentifikator 2 angegeben. Gleiches gilt für den Komponententyp kj, von dem die Verknüpfung zu ks ausgeht, sowie die verbundenen Komponententypen ow und ob. Durch die Einführung zusätzlicher Kontextidentifikatoren läßt sich also das Kontextproblem lösen.

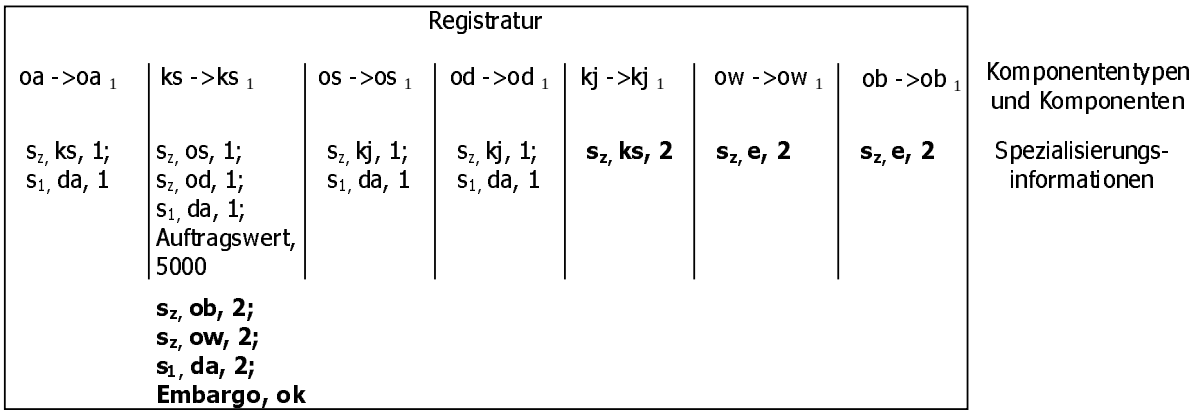


Abbildung 107: Registratur mit Spezialisierungsinformationen

10.1.4 Bewertung

Jetzt wird überprüft, ob mit Hilfe der vorgestellten Konzepte die schnelle Umsetzung von Workflow-Schemaänderungen möglich ist. Zur Veranschaulichung wird eine Erweiterung des Workflow-Schemas um eine Embargoprüfung vorgenommen. Nach der durchgeführten Detail- bzw. Standardprüfung findet ein Test statt, ob der Auftraggeber aus einem Embargoland stammt. In Abhängigkeit vom Ergebnis wird entweder die Bearbeitung fortgesetzt oder eine Absage verschickt. Die dementsprechend veränderte aspektelementorientierte Schemarepräsentation findet sich in Abbildung 108. Für die Verbindungen der neu hinzugekommenen Repräsentationselemente sind entsprechende Verbinder eingeführt worden.

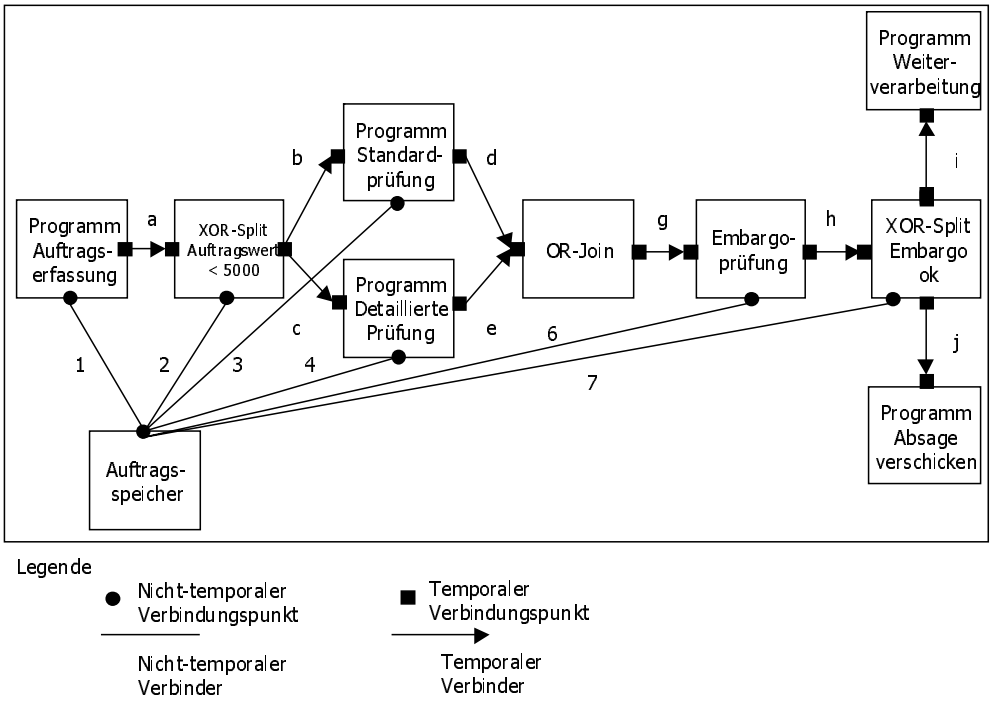


Abbildung 108: Erweiterte Schemarepräsentation

Es gilt diese Änderung des Workflow-Schemas in die komposite Anwendung einzubringen. Dazu ist zunächst die Menge der Komponententypen zu erweitern. Die zusätzlichen Komponententypen sind in Tabelle 10 dargestellt. Der Komponententyp oe dient der Durchführung der Embargoprüfung. Die Entscheidung über die Weiterleitung des Auftrages oder das Versenden einer Absage trifft erneut der Komponententyp ks, diesmal jedoch mit anderen Spezialisierungsinformationen. Schließlich muß noch ein Komponententyp für die Erstellung der Absage eingeführt werden. Er wird mit ob bezeichnet.

Aspektelementorientierte Schemarepräsentation	Komposite Anwendung
Embargoprüfung	oe
Gabelungselement XOR-Split (Entscheidung über Weiterleitung)	ks
Absage versenden	ob

Tabelle 10: Komponententypen der Erweiterung

In der nachfolgenden Abbildung 109 sind die zusätzlichen Komponententypen dargestellt. Der Komponententyp ob verschickt eine Absage und der Komponententyp ow ist für die Weiterverarbeitung zuständig. Die Komponententypen oe und ob enthalten die Schnittstelle s_z, mit der sie aktiviert werden bzw. ihren Nachfolger aktivieren in ein- und ausgehender Form. Die Schnittstelle s_1 ist eine ausgehende Schnittstelle, die zum Ansprechen der benötigten Daten dient.

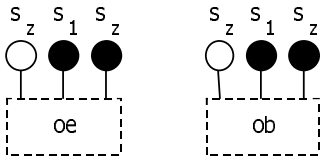


Abbildung 109: Komponententypen der Erweiterung

Der nächste Schritt ist die Umsetzung der Schemaänderung durch Einbindung der neu hinzugekommenen Komponenten. Hierzu müssen die Verknüpfungsinformationen mehrerer Komponenten verändert werden, wie in Tabelle 11 dargestellt. Der Wegfall des Verbinders f hat zur Folge, daß in den Verknüpfungsinformationen des Komponententyps kj die Verknüpfung mit ow gelöscht werden muß. Statt dessen gilt es den Verbinder g umzusetzen, weswegen eine Verknüpfung mit Komponententyp oe eingetragen werden muß. Das Repräsentationselement Embargoprüfung ist über den temporalen Verbinder h mit dem Repräsentationselement „Gabelungselement XOR-Split“ verbunden. Außerdem besteht der Verbinder 6 zum Repräsentationselement „Auftragsspeicher“. In die Verknüpfungsinformationen des Komponententyps oe muß daher eine Verknüpfung mit den Komponententypen ks und da eingetragen werden.

Das Repräsentationselement „Gabelungselement XOR-Split“ ist wiederum über temporale Verbinder, i und j, mit den Repräsentationselementen „Weiterverarbeitung“ und „Absage versenden“ verbunden. In die Verknüpfungsinformationen des Komponententyp ks müssen daher Verknüpfungen mit den Komponententypen ob und ow eingetragen werden. Zur Durchführung der Prüfung besteht vom Repräsentationselement „Gabelungselement XOR-Split“ außerdem der Verbinder 7 zum Repräsentationselement „Auftragsspeicher“. Auch dieser Verbinder muß durch entsprechende Verknüpfungsinformationen für die Komponententypen ks wiedergegeben werden. Eine Übersicht gibt Tabelle 11.

Komponententyp	Spezialisierungsinformationen
oa	$s_{z,}$ ks; $s_{1,}$ da
ks	$s_{z,}$ os; $s_{z,}$ od; $s_{1,}$ da; Auftragswert, 5000
os	$s_{z,}$ kj; $s_{1,}$ da
od	$s_{z,}$ kj; $s_{1,}$ da
kj	$s_{z,}$ oe
oe	$s_{z,}$ ks; $s_{1,}$ da
ks	$s_{z,}$ ob; $s_{z,}$ ow; s_{1} da; Embargo, ok
ob	$s_{z,}$ e
ow	$s_{z,}$ e

Tabelle 11: Spezialisierungsinformationen nach der Erweiterung

An der Komponente ks wird die Verwendung von unterschiedlichen Spezialisierungen besonders deutlich. Sie wurde bereits früher für die Entscheidung zwischen Detail- und Standardprüfung verwendet. Nun kann sie durch eine andere Spezialisierung für die Weiterverarbeitung in Abhängigkeit vom Ergebnis der Embargoprüfung eingesetzt werden.

Zum Abschluß wird auch noch das Ergebnis der Aufbereitung der Spezialisierungsinformationen für die registraturbasierte Spezialisierung in Abbildung 110 dargestellt. Deutlich sieht man, wie für die zweifache Verwendung des Komponententyps ks ein Wechsel des Kontextidentifikators durchgeführt wird. Auch hier ist der Eintrag für den Komponententyp da nicht dargestellt.

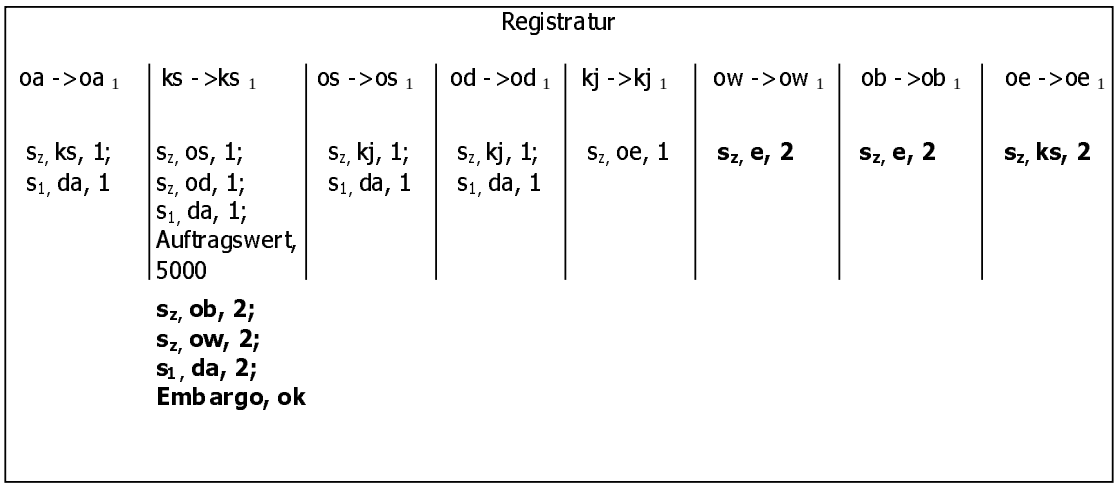


Abbildung 110: Behebung des Kontextproblems bei registraturbasierter Spezialisierung

Als Fazit kann gezogen werden, daß die hier vorgestellte Realisierung der aspektelementorientierten Schemarepräsentation die an sie gestellte Anforderung der schnellen Umsetzung von Workflow-Schemaänderungen erfüllt. So ist es möglich, Schemaänderungen schnell durch Anpassung der Verknüpfungs- und Spezialisierungsinformationen umzusetzen. Dies gilt sowohl für Schemaerweiterungen und –reduktionen als auch Topologie- und Parameteränderungen.

10.2 Realisierung des Workflow-Modell-Wörterbuchs

Nachdem im vorangegangenen Abschnitt gezeigt wurde, wie mit Hilfe kompositer Anwendungen die aspektelementorientierte Schemarepräsentation realisiert werden kann, wird jetzt die Realisierung des Workflow-Modell-Wörterbuchs beschrieben. Bereits bei oberflächlicher Betrachtung von Workflow-Modell-Wörterbuch und Registratur drängt sich die Frage auf, ob in der Registratur komponentenorientierter Rahmenwerke Informationen enthalten sind, die für das Workflow-Modell-Wörterbuch nutzbar sind.

Aus dieser Situation ergeben sich zwei Aufgaben. Zunächst muß geklärt werden, welche der Informationen in der Registratur und im Workflow-Modell-Wörterbuch verwendbar sind. In einem zweiten Schritt ist ein Integrationsmodell zu entwickeln, das die transparente Nutzung der in der Registratur enthaltenen Informationen ermöglicht.

10.2.1 Identifizierung verwendbarer Informationen

Ausgangspunkt für die Identifizierung äquivalenter Informationen ist die Registratur komponentenorientierter Rahmenwerke, die in Abbildung 111 dargestellt ist. Die Registratur gliedert sich nach Komponententypen. Jedem Komponententyp sind eine oder mehrere Komponentenbeschreibungen zugeordnet. Jede der Komponentenbeschreibungen steht für eine Komponente.

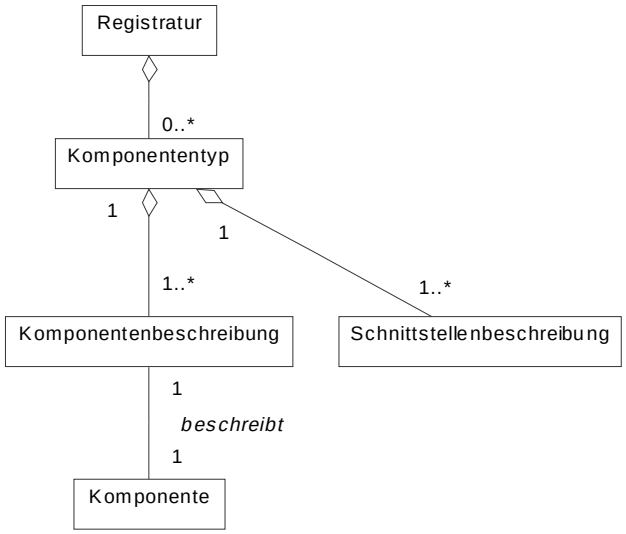


Abbildung 111: Registratur eines komponentenorientierten Rahmenwerks

Der für den Vergleich mit der Registratur des Workflow-Modells relevante Teil des Informationsmodells des Workflow-Modell-Wörterbuchs ist in Abbildung 112 dargestellt. Das Informationsmodell beschreibt die Objekttypen „Physisches Element“, „Dienstbeschreibung“ und „Ressourcenbeschreibung“. Zwischen den Objekttypen bestehen drei Beziehungen: Zwischen den Objekttypen „Physisches Element“ und „Dienstbeschreibung“ besteht die Unterstützungsbeziehung. Sie gibt wieder, durch welche Dienste ein bestimmtes physisches Element unterstützt wird. Zwischen den Objekttypen „Dienstbeschreibung“ und „Ressourcenbeschreibung“ besteht die Bereitstellungsbeziehung. Sie dient zur Wiedergabe der Information, welche Dienste durch welche Ressourcen erbracht werden.

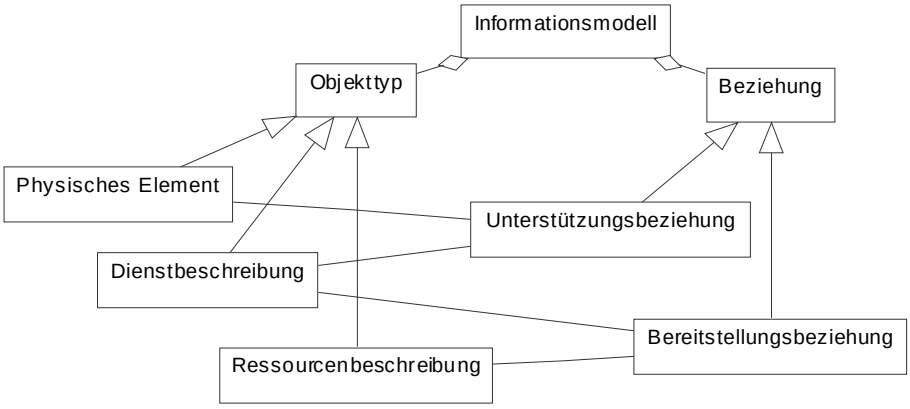


Abbildung 112: Ausschnitt des Informationsmodells des Workflow-Modell-Wörterbuchs

Nun wird für Objekttypen und Beziehungen des Workflow-Modell-Wörterbuchs untersucht, ob sich verwendbare Informationen in der Registratur komponentenorientierter Rahmenwerke befinden.

Betrachtet man komponentenorientierte Rahmenwerke aus der Perspektive des Workflow-Modell-Wörterbuchs, so stellen Komponenten Ressourcen dar. Die Komponentenbeschreibung in der Registratur komponentenorientierter Rahmenwerke enthält daher Informationen, die der Ressourcenbeschreibung im Informationsmodell des Workflow-Modell-Wörterbuchs zuzuordnen sind. Komponenten sind Ressourcen, die Dienste erbringen. Diese Dienste werden über die Schnittstellen der Komponente bereitgestellt. Daher können sich die Dienstbeschreibungen des Workflow-Modell-Wörterbuchs auf die Schnittstellenbeschreibungen der Registratur stützen.

Auch zwei Beziehungen des Informationsmodells finden sich in der Registratur wieder. Die Bereitstellungsbeziehung verbindet im Workflow-Modell-Wörterbuch Dienstbeschreibungen mit Ressourcenbeschreibungen. In der Registratur findet eine Zuordnung von Schnittstellenbeschreibungen zu Komponententypen und von diesen wiederum zu Komponentenbeschreibungen. Die Bereitstellungsbeziehung läßt sich also in der Registratur durch die Kette von Schnittstellenbeschreibungen zu Komponentenbeschreibungen wiedergeben.

10.2.2 Integrationsmodell

Auf der Basis dieser Überlegungen läßt sich folgendes Integrationsmodell für die Informationen im Workflow-Modell-Wörterbuch und der Registratur komponentenorientierter Rahmenwerke bilden. Es ermöglicht die transparente Nutzung der in der Registratur enthaltenen Informationen für die Zwecke des Workflow-Modell-Wörterbuchs. Dargestellt ist es in Abbildung 113.

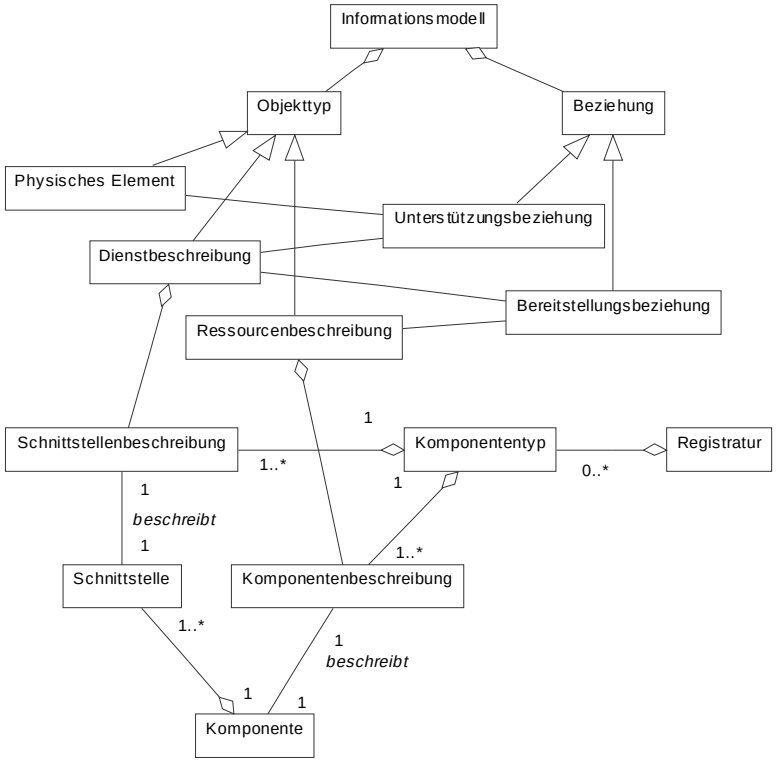


Abbildung 113: Integrationsmodell

10.3 Zusammenfassung

Ziele dieses Kapitels waren die Realisierung der aspektelementorientierten Schemarepräsentation und des Workflow-Modell-Wörterbuchs.

Das erste Ziel wurde durch eine Reihe von Lösungsschritten erreicht. Zunächst wurde eine Analyse durchgeführt, welche Bestandteile der aspektelementorientierten Schemarepräsentation durch welche Bestandteile von Komponentensystemen realisiert werden können. Auf den Ergebnissen dieser Analyse konnten dann Regeln für die konkrete Abbildung der korrespondierenden Bestandteile angegeben werden. So werden Repräsentationselemente durch aspektspezifische Komponenten einer kompositen Anwendung umgesetzt. Die Verbindungspunkte der Repräsentationselemente werden auf Schnittstellen abgebildet. Die Verbinder der kompositen Anwendung werden auf Verknüpfungsinformationen der kompositen Anwendung abgebildet.

Mit diesen Abbildungsregeln kann dann ein Verfahren für die konkrete Realisierung vorgestellt werden. Es besteht aus drei Realisierungsschritten. Im ersten Schritt werden Komponententypen bestimmt, die zur Realisierung der Repräsentationselemente verwendet werden sollen. Im zweiten Schritt werden Spezialisierungsinformationen für diese Komponententypen gebildet. Den dritten Schritt bildet schließlich die Anpassung der Spezialisierungsinformationen für das verwendete Spezialisierungsverfahren, also beispielsweise die registraturbasierte Spezialisierung.

Bei der Bildung von Spezialisierungsinformationen für die registraturbasierte Spezialisierung konnte das sogenannte Kontextproblem identifiziert werden. Zu seiner Behebung wurde eine Lösung angegeben. Eine abschließende Bewertung zeigt, daß die entwickelte Realisierung der aspektelementorientierten Schemarepräsentation zur Unterstützung aller Formen von Schemaänderungen in der Lage ist. So können sowohl Topologie- und Parameteränderungen als auch Schemaerweiterungen- und –reduktionen vorgenommen werden. Die Entwicklung des Workflow-Modell-Wörterbuchs konzentrierte sich auf die eigentliche Herausforderung, nämlich die Registratur komponentenorientierter Rahmenwerke zu integrieren. Daher wurde auf die detaillierte Darstellung der übrigen Bestandteile des Workflow-Modell-Wörterbuchs verzichtet. Das entwickelte Integrationsmodell erlaubt die transparente Nutzung von bereits in der Registratur vorhandenen Informationen in das Workflow-Modell-Wörterbuch. Auf diese Weise wird die Entstehung redundanter Informationen verhindert und eine effiziente Realisierung des Workflow-Modell-Wörterbuchs erreicht.

11 Inkrementelle Abbildung von Geschäftsprozessen

Eine wichtige Rahmenbedingung für die Unterstützung weitreichender Prozesse ist, daß Änderungen durch die größere Teilnehmerzahl häufiger auftreten. Schließlich ist jeder Prozeßteilnehmer eine potentielle Quelle von Änderungsanforderungen. In Kapitel 3 wurde hieraus die informationstechnische Anforderung abgeleitet, daß die Abbildung von Geschäftsprozeß- auf Workflow-Schemata inkrementell erfolgen muß. Ziel dieses Kapitels ist es daher, ein derartiges inkrementelles Verfahren für die Abbildung von Geschäftsprozeß- auf Workflow-Schemata zu erreichen.

Das Kapitel beginnt mit der Darstellung, welche Probleme bei der inkrementellen Abbildung von Geschäftsprozeß- auf Workflow-Schemata bestehen. Mit ihnen werden zunächst existierende Ansätze auf ihre Eignung untersucht. So ist die Abbildung von Geschäftsprozeßschemata auf Workflow-Schemata in einer Reihe von Arbeiten untersucht worden. Diese werden hier kurz vorgestellt und ihre Einschränkungen dargestellt, um die Notwendigkeit für die Entwicklung eines eigenen Verfahrens zu motivieren.

Die Analyse der existierenden Ansätze ergibt, daß ihre Einschränkungen in einem unvollständigen und nicht genügend detailliertem Raster zur Erfassung von Geschäftsprozeß- und Workflow-Modell begründet liegen. Die erste Herausforderung bei der Entwicklung des inkrementellen Abbildungsverfahrens liegt daher zunächst in der Bestimmung eines geeigneten Rasters für die Darstellung von Geschäftsprozeß- und Workflow-Modellen. Die zweite Herausforderung liegt in der Schaffung von Verknüpfungen zwischen den Rasterelementen von Geschäftsprozeß- und Workflow-Modell.

Zur Schaffung des Rasters werden drei Lösungskonzepte entwickelt. Es sind dies die Aspektseparation, Klassifizierungsstrukturen und Äquivalenzsicht. Das vierte Lösungskonzept, die Bildung von Abbildungsoperatoren, dient der Herstellung von Verknüpfungen zwischen Geschäftsprozeß- und Workflow-Modell. Diese Lösungskonzepte sind auf der Modellebene angesiedelt und werden dann zur Durchführung der Abbildungen auf der Schemaebene verwendet. Die Abbildung auf der Schemaebene besteht aus drei Schritten. Es sind dies die Schaffung eines Äquivalenzschemas, eines Workflow-Gerüsts und die Ergänzung dieses Gerüsts. Den Abschluß des Kapitels bildet die Bewertung der erreichten Ergebnisse.

11.1 Problemstellung

Die Abbildung von Geschäftsprozeß- auf Workflow-Schemata bedeutet, die nach Vorgabe eines Geschäftsprozeßmodells dargestellten Informationen des Geschäftsprozeßschemas in ein Workflow-Schema zu überführen, das nach Maßgabe des Workflow-Modells gestaltet ist. Das Problem ist dabei, daß Geschäftsprozeß- und Workflow-Modell für unterschiedliche Ziele gestaltet worden sind, die sich in unterschiedlichen Abstraktionen niederschlagen.

Geschäftsprozeßmodelle haben eine betriebswirtschaftliche Zielsetzung. Mit ihnen werden Geschäftsprozesse erfaßt und einer betriebswirtschaftlichen Optimierung zugänglich gemacht. Workflow-Modelle haben die Optimierung der informationstechnischen Umsetzung zum Ziel. Daher unterscheiden sich die von Geschäftsprozeß- und Workflow-Modell gebildeten Abstraktionen, sprich die Elemente des Geschäftsprozeß- und des Workflow-Modells. Es überrascht daher nicht, daß auf der Basis von Geschäftsprozeß- und Workflow-Modellen gebildete Geschäftsprozeß- und Workflow-Schemata den gleichen Sachverhalt unterschiedlich und mit unterschiedlicher Detaillierung darstellen. Soll eine Abbildung von Geschäftsprozeß- auf Workflow-Schemata erfolgen, so müssen diese Differenzen in der Darstellung überwunden werden. Es gilt eine Abbildung von den Abstraktionen des Geschäftsprozeßmodells auf das Workflow-Modell zu schaffen. Veranschaulicht sind diese Zusammenhänge in Abbildung 114.

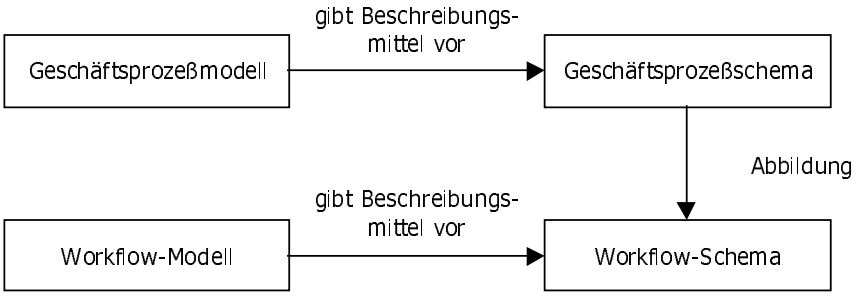


Abbildung 114: Abbildung von Geschäftsprozeß auf Workflow-Schemata

Eine inkrementelle Abbildung bedeutet, daß nach der erstmaligen Abbildung des Geschäftsprozeß- auf das Workflow-Schema Änderungen des Geschäftsprozeßschemas keine erneute Gesamtabbildung mehr erforderlich machen. Statt dessen kann auf Grund der Änderung des Geschäftsprozeßschemas die Menge der Workflow-Schemaelemente bestimmt werden, die angepaßt werden müssen, um die Veränderung des Geschäftsprozeßschemas wiederzugeben. Dies bedeutet, die nach der erstmaligen Abbildung eines Geschäftsprozeßschemas auf ein Workflow-Schema nachfolgende Änderungen so durchzureichen, daß die Teile des Workflow-Schemas, die nicht von der Änderung betroffen sind, unbeeinflußt bleiben. Veranschaulicht ist dies in Abbildung 115. Der durch die Änderungen betroffene Bereich ist jeweils schraffiert. Bei der links dargestellten Gesamtabbildung muß immer das vollständige Geschäftsprozeßschema abgebildet werden. Bei der rechts dargestellten inkrementellen Abbildung kann von den veränderten Bestandteilen des Geschäftsprozeßschemas auf die zu verändernden Bestandteile des Workflow-Schemas geschlossen werden.

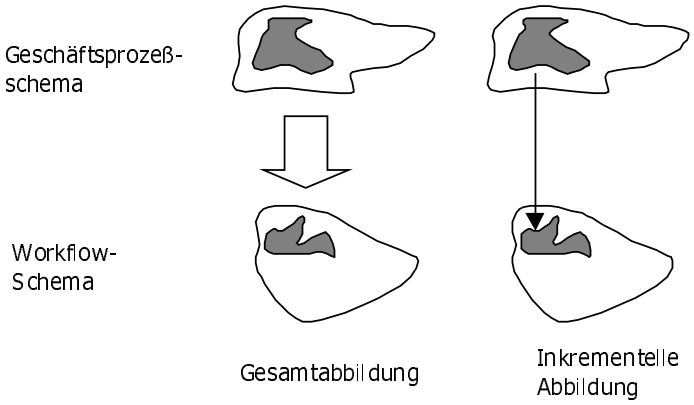


Abbildung 115: Vergleich zwischen Gesamtabbildung und inkrementeller Abbildung

Um eine inkrementelle Abbildung zu erreichen, sind zwei Probleme zu lösen. Dargestellt ist dies in Abbildung 116. Zuerst sind sowohl im Geschäftsprozeß- als auch im Workflow-Modell Teile zu identifizieren, die unabhängig voneinander verändert werden können und sich dabei nicht gegenseitig beeinflussen. Es gilt also die Auswirkung von Änderungen so klein wie möglich zu halten. Um dies zu erreichen, müssen Geschäftsprozeß- und Workflow-Modell einem Raster unterworfen werden, das es erlaubt, die Modellteile zu identifizieren, die unabhängig voneinander verändert werden können.

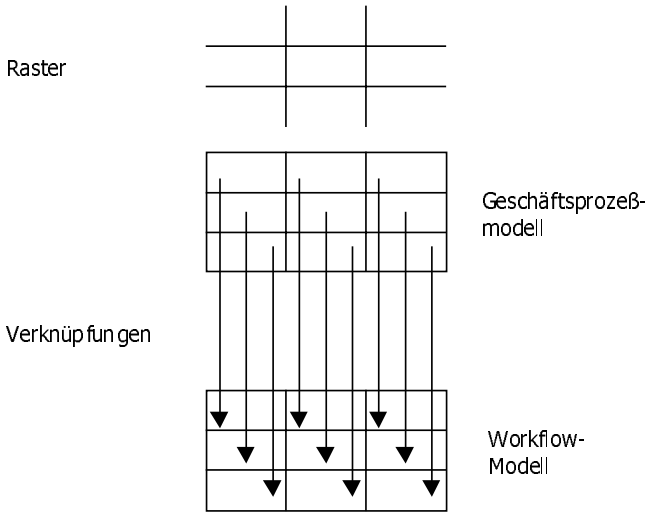


Abbildung 116: Raster und Verknüpfungen bei der inkrementellen Abbildung

Das zweite Problem ist die Schaffung von Verknüpfungen vom Geschäftsprozeßmodell zum Workflow-Modell, die eine Identifikation der Elemente des Workflow-Schemas erlaubt, die von einer Änderung des Geschäftsprozeßschemas betroffen sind. Dabei kommt das geschaffene Raster zum Einsatz. Die von dem Raster geschaffenen Rasterelemente bilden den Anfangs- und Endpunkt der Verknüpfung. Auf der Basis dieser Verknüpfung können letztendlich Abbildungsoperatoren definiert werden.

Auf der Basis von derart gerasterten und miteinander verknüpften Geschäftsprozeß- und Workflow-Modellen kann dann eine inkrementelle Abbildung stattfinden. Veranschaulicht ist dies in Abbildung 117. Die Rasterung von Geschäftsprozeß- und Workflow-Modell wird auf das Geschäftsprozeß- bzw. Workflow-Schema übertragen. Sie erlaubt es, die unabhängig voneinander veränderbaren Bestandteile des Geschäftsprozeß- und Workflow-Schemas zu identifizieren. Ebenso können die auf Modellebene definierten Verknüpfungen verwendet werden, um die Abbildung der veränderten Bestandteile des Geschäftsprozeßschemas durchzuführen.

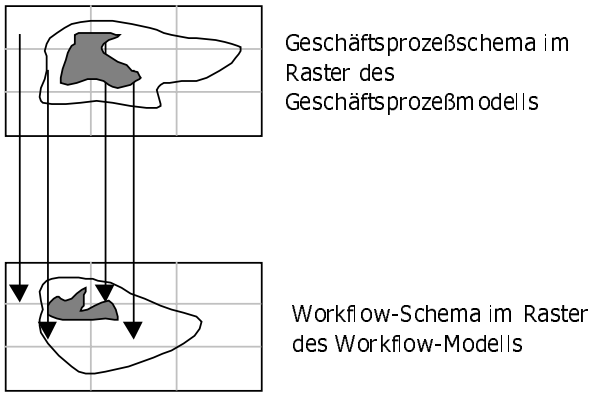


Abbildung 117: Durchführung einer inkrementellen Abbildung

11.2 Existierende Ansätze

Zunächst werden existierende Ansätze auf ihre Eignung für die inkrementelle Abbildung von Geschäftsprozeß- auf Workflow-Schemata hin untersucht. Kriterien sind dabei die Bereitstellung eines Rasters zur Darstellung von Geschäftsprozeß- und Workflow-Modell sowie die Gestaltung der Verknüpfungen zwischen den Bestandteilen des Rasters.

11.2.1 Abbildung von SOM auf BusinessFlow-Schemata

Dieses Verfahren, in [Kreu96] vorgestellt, bildet Schemata der zweiten Ebene des SOM-Ansatzes auf Workflow-Schemata des Systems BusinessFlow ab. Weder die Sicht der Unternehmensziele noch die Ressourcen des Unternehmens finden Eingang in das Abbildungsverfahren. Ausgangspunkt sind SOM-Prozeßschemata, von denen ein hinreichender Detaillierungsgrad erwartet wird. Diese Anforderung wird jedoch nicht weiter spezifiziert, mit der Ausnahme, daß Kontrollflußkonstrukte besonders dokumentiert werden.

11.2.1.1 Vorgehen

Das Verfahren beschränkt sich auf die zweite Ebene des SOM-Modells. D.h. es unterstützt nur die Abbildung von Schemata, die Elemente der zweiten Ebene des SOM-Modells enthalten. Für die Bildung von Geschäftsprozeßschemata werden keine besonderen Vorgaben gemacht. Dies bedeutet, daß es kein Kriterium dafür gibt, wann eine hinreichend detaillierte Darstellung des Geschäftsprozesses im Schema erreicht ist. Betrachtet werden zudem nur Schemata auf der untersten Zerlegungsebene mit dem höchsten Detaillierungsgrad. Abstraktere Ebenen werden außer Acht gelassen. Das Vorgehen dieses Ansatzes wird in Abbildung 118 veranschaulicht. In ihm sind die einzelnen Schritte mit Nummern versehen, die im folgenden detailliert betrachtet werden.

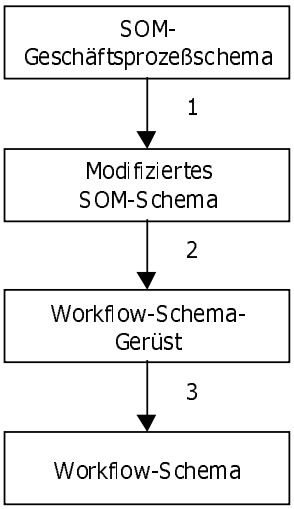


Abbildung 118: Vorgehensweise bei der Ableitung von BusinessFlow-Workflow-Schemata

1. Der erste Schritt ist die Schaffung eines modifizierten SOM-Schemas. Für die spätere Schaffung eines Workflow-Schemas irrelevante Informationen werden entfernt. Im wesentlichen werden nur Aufgaben und Ereignisse betrachtet, während die strukturorientierten Anteile des SOM-Schemas unterdrückt werden. Durch Hilfskonstrukte werden Umweltaufgaben und Umweltereignisse übertragen. Das modifizierte Prozeßschema besteht nur aus Aufgaben und Ereignissen und muß eventuell in Teilprozesse zerlegt werden, um abgebildet werden zu können. Es gibt jedoch keine allgemeinen Regeln, sondern nur Konstellationen, die empfehlenswerte Zerlegungen darstellen. Die Entscheidung hierüber läßt sich jedoch nur durch Kenntnis des Prozesses treffen.

2. Die eigentliche Abbildung ist auf der Modellebene beschrieben und gestaltet sich wegen der starken Reduktion einfach. Die Abbildungsregeln hierzu sind in Abbildung 119 dargestellt. So werden betriebliche Aufgaben auf Aktionen innerhalb von Aktivitäten in BusinessFlow abgebildet. Diskurs- und Umweltereignisse werden auf Variablen der Aktivitäten, die den Kontrollfluß steuern übertragen. Personen oder Organisationseinheiten werden bei der Abbildung nicht berücksichtigt. Die Übertragung von Kontrollflußkonstrukten bedarf der Interpretation durch den Modellierer. Als Ergebnis erhält man ein Workflow-Schema-Gerüst, das durch den Modellierer manuell ergänzt werden muß.

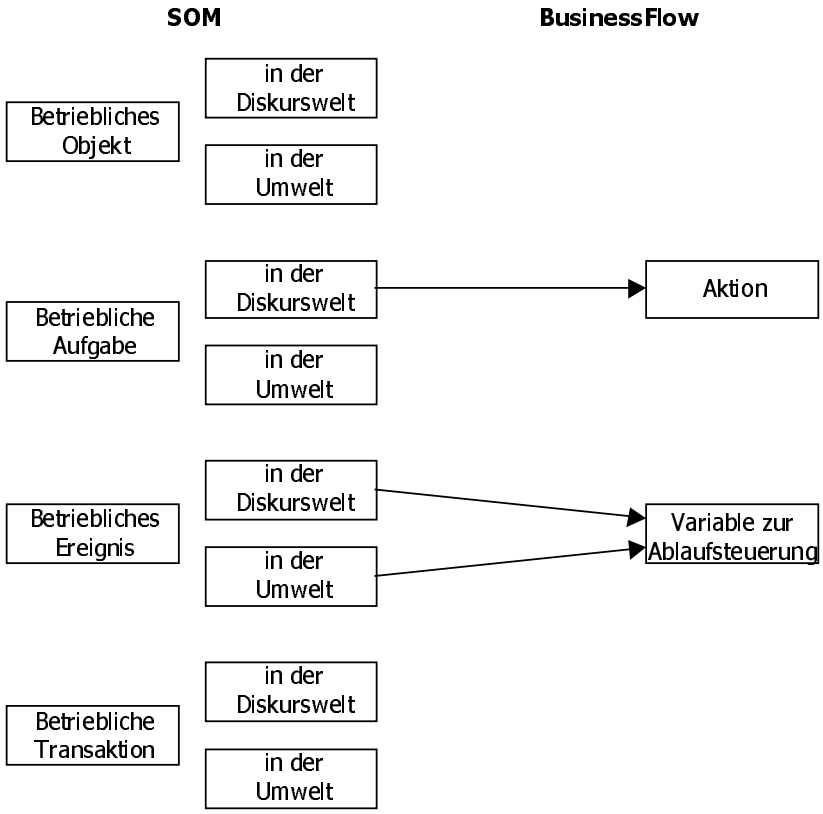


Abbildung 119: Modellobjektkorrespondenzen in SOM und BusinessFlow

3. Der dritte Schritt ist die Durchführung manueller Ergänzungen. Diese sind jedoch nicht näher erläutert. Sie müßten jedoch zumindest Spezifikation von Verfahren oder Algorithmen enthalten, die für die Durchführung der Aktionen verwendet werden. Ebenfalls müssen Ablaufsteuerungsinformationen ergänzt werden.

11.2.1.2 Bewertung

Das von diesem Verfahren entwickelte Raster ist sehr unvollständig. So wird nur die zweite Ebene des SOM-Ansatzes betrachtet. Auch die Strukturinformationen des SOM-Modells werden nicht erfaßt, obwohl diese die Grundlage für die Verknüpfung mehrerer Prozesse bilden. Ebenfalls nicht erfaßt wird die in SOM erfolgende Differenzierung zwischen Entscheidungs- und Transformationsaufgaben sowie Transaktions- und objektinternen Ereignissen.

Das verwendete Raster ist sehr grob und es existiert nur eine geringe Zahl von Verknüpfungen. Daher sind in hohem Maße Interpretationsvorgänge durch den Modellierer notwendig. Für diese sind keine Regeln angebbar, so daß die Kenntnis der Intention und der Details des Prozeßschemas für die Abbildungsqualität entscheidend ist. Als Fazit ist daher zu ziehen, daß wegen des unvollständigen und groben Rasters sowie der geringen Zahl von Verknüpfungen eine inkrementelle Abbildung durch das Verfahren nicht erreichbar ist.

11.2.2 Abbildung von SOM auf WorkParty

Das in [Lang96] beschriebene Verfahren verzichtet ebenfalls auf eine Nutzung von Informationen aus der ersten und dritten Ebene des SOM-Modells. Nur die zweite Ebene des SOM-Ansatzes wird berücksichtigt. An die abzubildenden Schemata werden keine expliziten Modellierungskonventionen gestellt.

11.2.2.1 Vorgehen

Das Verfahren besitzt zwei Schritte, die in Abbildung 120 dargestellt sind.

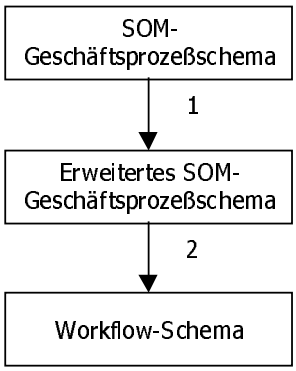


Abbildung 120: Vorgehensweise bei der Ableitung von WorkParty-Workflow-Schemata

1. Im ersten Schritt der Abbildung wird ein erweitertes SOM-Geschäftsprozeßschema gebildet. Zunächst werden dazu Kontrollflußkonstrukte im SOM-Schema identifiziert und eine Modellierung personeller Ressourcen vorgenommen. Auch wird dabei eine Zerlegung in Teilprozesse vorgenommen.

2. Die Abbildung erfolgt auf der Modellebene und verwendet die in Abbildung 121 dargestellten Modellobjektkorrespondenzen. Betriebliche Objekte der Diskurswelt werden auf Rollen abgebildet. Die Zuweisung konkreter Personen geschieht durch die externe Organisationsmodellierung. Dabei wird erwartet, daß eine Person alle Aufgaben auszuführen vermag, die einem betrieblichen Objekt zugeordnet sind. Die Abbildung der betrieblichen Objekte auf organisatorische Rollen in WorkParty setzt voraus, daß diese auch konkret vorhanden sind. Mit diesen Objekten assoziierten Aufgaben wird diese Rolle als personeller Aufgabenträger zugeordnet. Abweichend vom SOM-Modell werden Umweltobjekte nicht abgebildet. Betriebliche Aufgaben in SOM werden auf Tätigkeiten in WorkParty abgebildet. Die in SOM stattfindende Differenzierung betrieblicher Aufgaben anhand ihrer Automatisierbarkeit wird auf die ähnlich erfolgende Unterscheidung in WorkParty übernommen. So werden automatisierte Aufgaben auf programmgeführte Tätigkeiten abgebildet, teilautomatisierte auf programmgestützte und nicht automatisierte auf Leertätigkeiten. Es handelt sich dabei um Tätigkeiten, die im WfMS nicht unterstützt werden, sondern lediglich verwaltet werden. Die Erweiterungen des SOM-Modells zur Darstellung von Kontrollflüssen werden bei der Abbildung eines dementsprechend erweiterten SOM-Schemas zur Abbildung auf Verzweigungen, Synchronisationen usw. genutzt. Betriebliche Ereignisse und Transaktionen werden auf Ablaufkonstrukte abgebildet. Umweltereignisse und Umwelttransaktionen werden auf Leertätigkeiten in WorkParty abgebildet. Diese müssen durch den Benutzer manuell beendet werden.

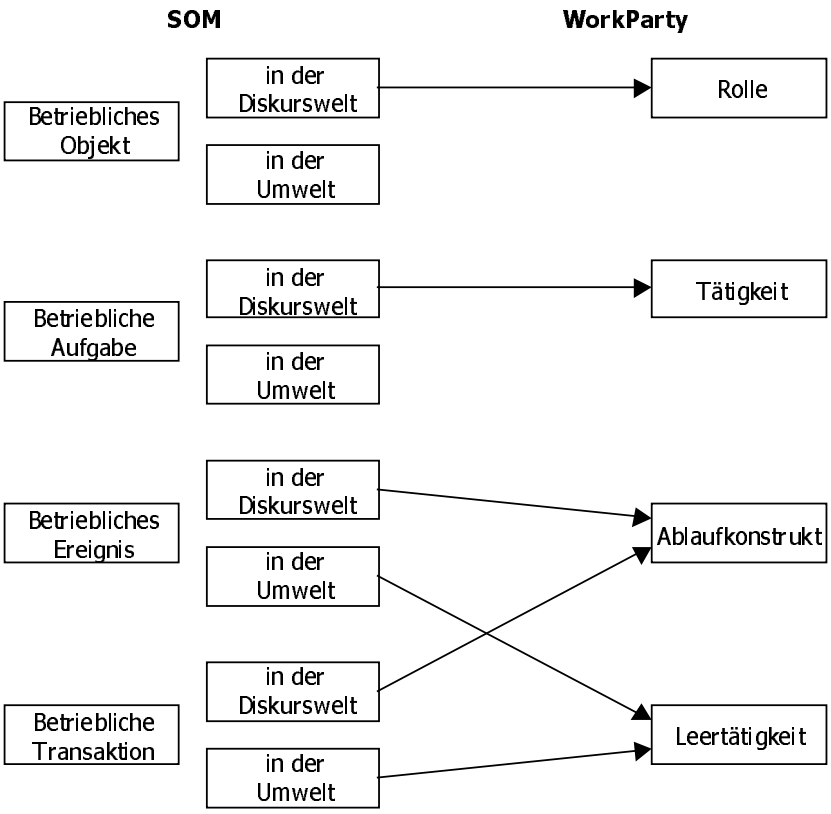


Abbildung 121: Modellobjektkorrespondenzen in SOM und WorkParty

11.2.2.2 Bewertung

Auch das Raster dieses Abbildungsverfahrens ist unvollständig, da es nur Informationen der zweiten Ebene des SOM-Modells berücksichtigt. Dennoch sind gegenüber dem vorangegangenen Ansatz einige Fortschritte erkennbar. So wird durch die Einführung zusätzliche Elemente zur Darstellung des Kontrollflusses das Raster deutlich feiner als beim letzten Ansatz.

Dies schlägt sich darin nieder, daß deutlich mehr Konstrukte ohne Interpretation, d.h. manuellen Eingriff des Modellierers abgebildet werden können. Allerdings wird auch in diesem Ansatz kein ausreichend feines Raster erreicht wird, so daß eine inkrementelle Abbildung prinzipiell nicht möglich erscheint.

11.2.3 Abbildung von SOM-Schemata auf FlowMark

Ausgangsbasis des in [Ambe95], [Ambe96a] und [Luko96] beschriebenen Verfahrens ist wie bei den zuvor behandelten Abbildungsverfahren die feinste Zerlegungsstufe der Ablaufsicht der Prozesse. Ebenfalls wird wiederum nur die zweite Ebene betrachtet. Es werden jedoch keine konkreten Kriterien angegeben, wann eine geeignet feine Zerlegung des SOM-Schemas für die Zwecke des Abbildungsverfahrens vorliegt. Auch sonst werden keine konkreten Kriterien an ein SOM-Schema gestellt, das abgebildet werden soll.

11.2.3.1 Vorgehen

Das Verfahren besteht aus zwei Schritten, wie in Abbildung 122 dargestellt. Im ersten wird das SOM-Schema in Teilschemata, sogenannte SOM-Workflows zerlegt. Der zweite Schritt ist die Abbildung der Teilschemata auf Flow-Mark-Schemata.

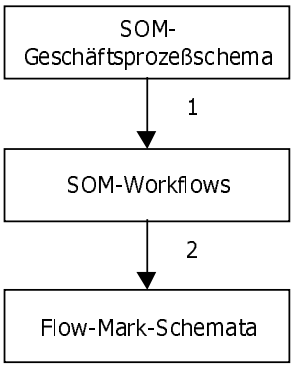


Abbildung 122: Vorgehensweise bei der Ableitung von BusinessFlow-Workflow-Schemata

1. Im ersten Schritt erfolgt die Aufteilung des SOM-Schemas. Es werden Kriterien angegeben, die entscheiden helfen, ob eine Zerlegung notwendig ist. Die Durchführung der Zerlegung wird durch Regeln beschrieben. Erstes Kriterium ist das Vorhandensein von einer oder mehreren Startaufgaben. Daher werden alle Aufgaben eines SOM-Schemas, das direkt oder transitiv von einer oder mehreren Startaufgaben erreichbar ist, in einem Teilschema zusammengefaßt. Das zweite Kriterium ist das Auftreten einer unterschiedlichen Zahl von Aufgabenausprägungen. Besitzen zwei Aufgaben eine unterschiedliche Anzahl von Ausprägungen, werden die dazugehörigen Aufgaben unterschiedlichen Teilschemata zugeordnet. Als Ergebnis des ersten Abbildungsschrittes erhält man sogenannte SOM-Workflows.

2. Die SOM-Workflows sind Ausgangspunkt für den zweiten Schritt des Verfahrens. Jeder SOM-Workflow wird separat abgebildet. Die Regeln für die Abbildungen sind auf Modellebene angegeben und in Abbildung 123 dargestellt.

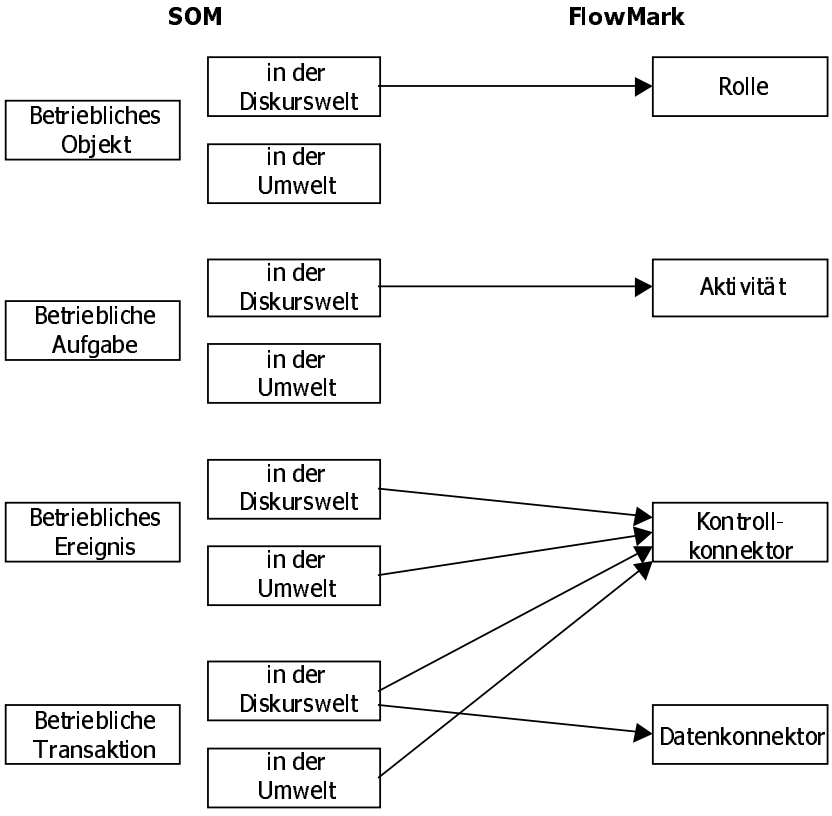


Abbildung 123: Modellobjektkorrespondenzen in SOM und FlowMark

Betriebliche Objekte werden auf Rollen abgebildet. Aufgaben werden auf Programm- und Prozeßaktivitäten, Aufgabenzyklen auf Prozeßaktivitäten mit einer Endbedingung abgebildet. Ist der Automatisierungsgrad für die Vorgangssteuerung, -auslösung und -durchführung sowie die Innensicht der Aufgabe bekannt, kann durch die angegebenen Regeln eine Abbildung automatisch durchgeführt werden. Mit Diskursweltaufgaben verbundene Umweltaufgaben werden durch ein Hilfskonstrukt ersetzt. Es handelt sich dabei um eine Kombination aus einem objektinternen Ereignis und einem Umweltereignis. Die Unterscheidung zwischen echten und unechten Umweltobjekten werden dabei ignoriert, so daß Objekte, die zwar zum Unternehmen zugehörig sind, aber nicht der Diskurswelt angehören, unberücksichtigt bleiben. Mit der Begründung, daß parallel zum Kontrollfluß auch Daten ausgetauscht werden, werden Transaktionen in Kontroll- und Datenkonnektoren umgesetzt. Ebenfalls als Kontrollkonnektor werden Ereignisse wiedergegeben. Die Datenflüsse zwischen einzelnen Aufgaben eines betrieblichen Objektes werden jedoch mit dem Argument nicht wiedergegeben, diese würden einen gemeinsamen Speicher besitzen, so daß ein Datenaustausch unnötig sei. Interessant ist dieser Umstand, da hierbei das Verständnis zu Grunde liegt, daß ein betriebliches Objekt einem Anwendungssystem entspricht. Dies widerspricht aber der oben durchgeführten Abbildung von betrieblichen Objekten auf Rollen.

11.2.3.2 Bewertung

Vom Raster dieses Verfahrens werden nur Informationen der verhaltensorientierten Sicht erfaßt. Andere Informationen der zweiten Ebene von SOM oder Informationen der ersten und dritten Ebene werden nicht berücksichtigt. Das Raster bleibt daher unvollständig. Der Detaillierungsgrad für das Raster schwankt. So ist es für viele Informationen relativ grob, andere Informationen wie beispielsweise Aufgaben werden jedoch sehr fein dargestellt. Daher ist die Menge der Verknüpfungen, die für eine inkrementelle Abbildung nutzbar wären, sehr unterschiedlich. Insgesamt wird daher die Anforderung der inkrementellen Abbildung nicht erfüllt.

11.2.4 Fazit

Keines der untersuchten Verfahren erreicht die erstrebte inkrementelle Abbildung von Geschäftsprozeß- auf Workflow-Schemata. Bei der Untersuchung der existierenden Ansätze konnten zwei Problembereiche identifiziert werden. So ist das von den existierenden Ansätzen geschaffene Raster unvollständig. Dies hat zur Folge, daß die Informationen im Geschäftsprozeß- und im Workflow-Modell nicht vollständig erfaßt werden. Zusätzlich ist das Raster nicht fein genug, um die Grundlage für eine inkrementelle Abbildung zu bilden. Es erlaubt nicht die erforderliche eindeutige Zuordnung von Elementen des Geschäftsprozeßmodells zu denen des Workflow-Modells. Dementsprechend kann auch nicht für eine geeignete Menge von Verknüpfungen gesorgt werden, die für eine inkrementelle Abbildung erforderlich ist. Die Automatisierbarkeit der Verfahren ist außerdem sehr gering, da in hohem Maße Interpretationen durch den Modellierer in die Abbildungen einfließen. Der daraus erwachsende hohe Aufwand führt zur Gefahr, daß sich in der Praxis Geschäftsprozeß- und Workflow-Schema nach kurzer Zeit voneinander lösen und unabhängig weiterentwickeln.

Insgesamt weisen die existierenden Verfahren nur ein sehr schmales methodisches Fundament auf. Es werden mehr oder minder ad-hoc methodenspezifische Zusammenhänge identifiziert und daraus Abbildungsregeln abgeleitet. Durch diese methodischen Schwächen kommt es verstärkt zur Notwendigkeit von schemaspezifischen Modifizierungen und Ergänzungen, die keiner Systematik unterworfen sind.

11.3 Beispielschema

Bevor die Entwicklung eines neuen Verfahrens begonnen wird, wird ein kleines Geschäftsprozeßschema zur Veranschaulichung präsentiert. Es ist in der SOM-Notation dargestellt, die in Kapitel 2 beschrieben wurde.

In Abbildung 124 ist der für den darzustellenden Geschäftsprozeß relevante Ausschnitt des hypothetischen Unternehmensplans dargestellt. An oberster Stelle soll das Ziel der Gewinnmaximierung stehen. Als Strategie wird verfolgt, die Durchlaufzeit für die Bearbeitung von Aufträgen zu minimieren.

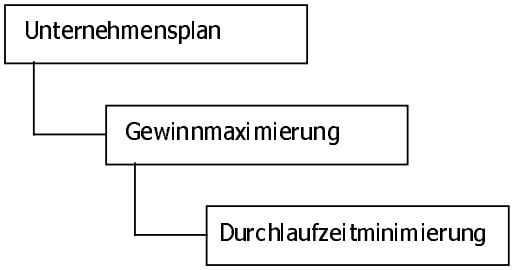


Abbildung 124: Unternehmensplan (Ausschnitt)

Teil der Strategie zur Durchlaufzeitminimierung ist, die Durchlaufzeit bei der Auftragsbearbeitung zu minimieren. Dazu wird die Bearbeitung der Aufträge als Prozeß organisiert und optimiert. Die oberste Ebene der prozeßorientierten Darstellung ist in Abbildung 125 wiedergegeben. Sie besteht aus einem Interaktionsdiagramm mit den Umweltobjekten Kunde und Fertigung und dem betrieblichen Objekt Vertrieb. Der Kunde gibt einen Auftrag an den Vertrieb. Der Vertrieb erfaßt und prüft den Auftrag und übergibt ihn nach der Prüfung an die Fertigung.

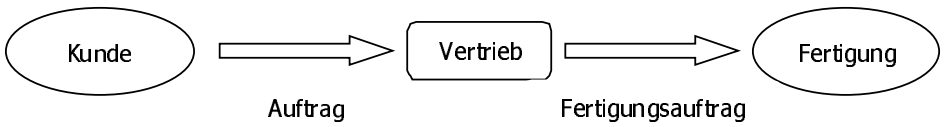


Abbildung 125: Interaktionsdiagramm des Geschäftsprozesses

Dieses Interaktionsdiagramm weist aber noch nicht die für die Prozeßgestaltung ausreichende Detaillierung auf, weswegen eine Detaillierung durchgeführt wird. Dazu wird das betriebliche Objekt Vertrieb in die zwei betrieblichen Objekte, nämlich Auftragserfassung und –prüfung zerlegt. Dies ist in Abbildung 126 dargestellt.

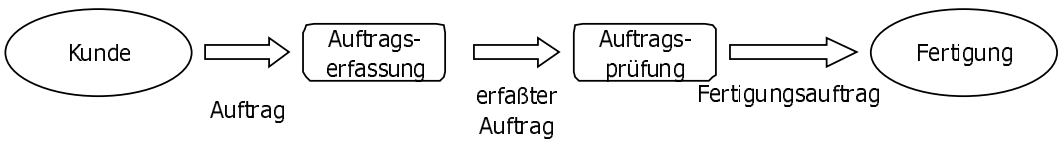


Abbildung 126: Detailliertes Interaktionsdiagramm des Geschäftsprozesses

Mit diesem Schritt ist eine ausreichende Detaillierung erreicht, um ein Vorgangs-Ereignis-Schema zu bestimmen. Es ist in Abbildung 127 dargestellt. Den betrieblichen Objekten und Umweltobjekten werden Aufgaben zugeordnet, die über Vor- und Nachereignisse verbunden sind. So ist dem Umweltobjekt Kunde die Aufgabe „Auftrag erteilen“ zugeordnet. Der Auftragserfassung ist die Aufgabe „Auftrag erfassen“ zugeordnet. Die Auftragsprüfung hat zwei Aufgaben: die Detailprüfung und die Standardprüfung. Die Entscheidung hierüber wird in Abhängigkeit davon getroffen, ob der Auftragswert DM 5000 überschreitet oder nicht. In Abhän-

gigkeit von den Ergebnissen der Prüfungen wird die Ausführung fortgesetzt. Ist die Auftragsprüfung ok, so wird der Auftrag an die Fertigung weitergegeben, wo die Bearbeitung fortgesetzt wird (dieser Schritt ist nicht näher detailliert dargestellt). Ist dies nicht der Fall, wird eine Absage versandt.

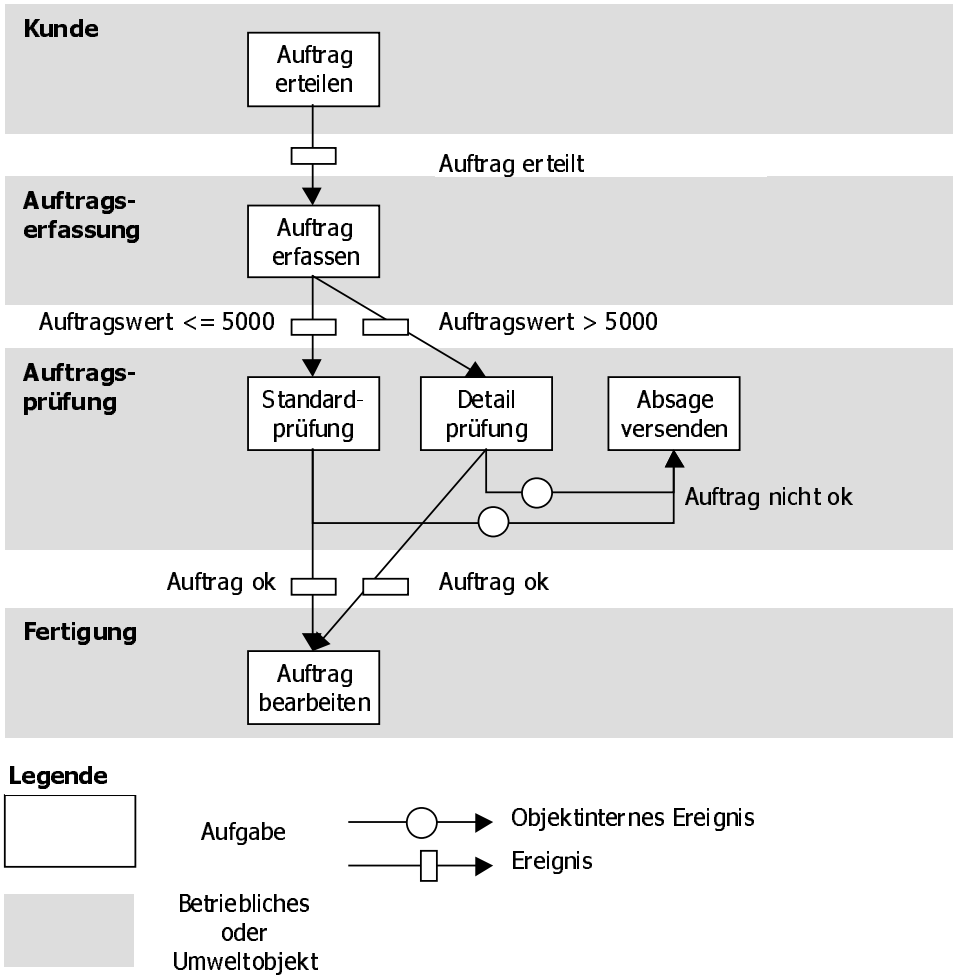


Abbildung 127: Vorgangs-Ereignis-Schema

Eine wichtige Ergänzung ist die Darstellung der personellen und maschinellen Aufgabenträger und ihre Zuordnung zu den obigen Aufgaben (Sie ist hier nicht dargestellt). Die personellen Aufgabenträger sind in Abbildung 128 dargestellt. So setzt sich die Organisationseinheit Vertrieb aus den Rollen Sachbearbeiter und Abteilungsleiter zusammen. Die Rolle Sachbearbeiter ist den Aufgaben „Auftrag erfassen“, „Standardprüfung“ und „Auftrag bearbeiten“ zugewiesen. Die Rolle Abteilungsleiter ist der Aufgabe „Detailprüfung“ zugewiesen. Der Aufgabe „Absage versenden“ ist kein personeller Aufgabenträger zugeordnet, da diese Aufgabe vollautomatisch erfolgt.

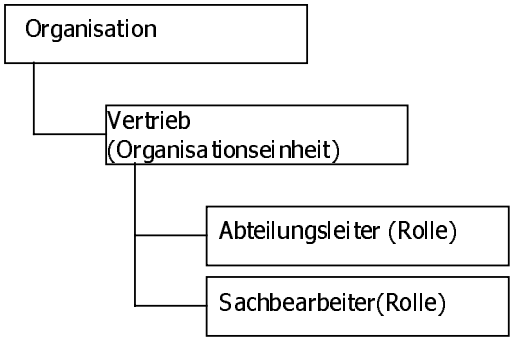


Abbildung 128: Darstellung der personellen Aufgabenträger

11.4 Lösungsansatz

Die Untersuchung existierender Ansätze zeigte nicht nur deren Einschränkungen auf, sondern brachte auch die zugrundeliegenden Problembereiche zu Tage:

Der erste Problembereich ist die Gestaltung des Rasters zur Darstellung von Geschäftsprozeß- und Workflow-Modell, der zweite die Gestaltung der Verknüpfungen zwischen Geschäftsprozeß- und Workflow-Modell. Zunächst wird daher ein geeignet gestaltetes Raster zur Erfassung von Geschäftsprozeß- und Workflow-Modell geschaffen. Das Raster zur Darstellung der Informationen in Geschäftsprozeß- und Workflow-Modell ist so zu konzipieren, daß zwei Anforderungen erfüllt werden. Die erste Anforderung ist, daß alle Informationen sowohl im Geschäftsprozeß- als auch im Workflow-Modell erfaßt werden müssen. Die zweite Anforderung ist, daß das Raster fein genug sein muß, um genügend kleine Änderungssphären sicherzustellen.

Eine Übersicht der drei Lösungsschritte, mit denen diese Anforderungen erfüllt werden, gibt Abbildung 129. Es handelt sich um die Aspektseparation, Klassifizierungsstrukturen, die Äquivalenzsicht und Abbildungsoperatoren.

1. Der erste Schritt zur Schaffung eines Rasters ist die Aspektseparation. Durch die Aspektseparation kann der Informationsgehalt der beiden Modelle beschrieben werden: Es können die zur Abbildung verfügbaren Informationen, aber auch die noch zu ergänzenden Informationen identifiziert werden. Zusätzlich werden durch die Aspektseparation Änderungssphären geschaffen, die einen ersten Schritt zur Eingrenzung von Abbildungen darstellen.

2. Eine Verfeinerung des Rasters wird durch Klassifizierungsstrukturen erreicht. Durch Klassifizierungsstrukturen werden die durch die Aspektseparation identifizierten Teilmodelle nochmals zerlegt und für das Abbildungsverfahren erschlossen.

3. Durch die Anwendung der Aspektseparation und der Klassifizierungsstrukturen entsteht ein Raster aus aspektseparaten Modellelementen. Es muß aber auch der Fall behandelt werden, daß semantisch äquivalente Informationen in beiden Modellen unterschiedlich strukturiert werden. Damit unter diesen Umständen die Schaffung von Verknüpfungen möglich ist, muß eine Äquivalenzsicht geschaffen werden. Diese ist semantisch zum Geschäftsprozeßmodell und strukturell zum Workflow-Modell äquivalent.

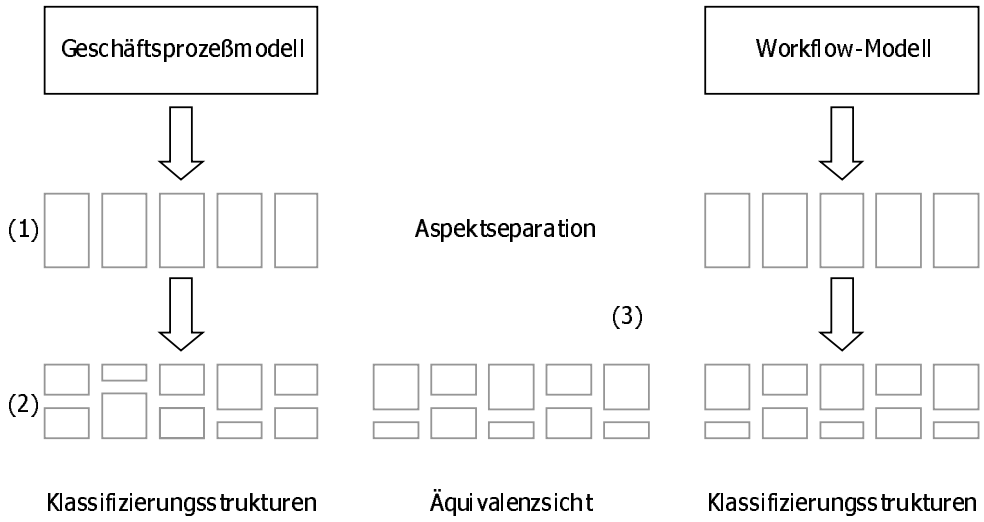


Abbildung 129: Lösungsschritte bei der Bestimmung des Rasters

11.4.1 Aspektseparation

Zunächst muß eine Rastereinteilung gefunden werden, mit der es möglich ist, unterschiedliche Arten von Informationen im Geschäftsprozeß- bzw. Workflow-Modell zu unterscheiden. Dabei wird auch aufgedeckt, welche Informationen im Geschäftsprozeß- und im Workflow-Modell enthalten sind und insbesondere welche Informationen in einem der beiden fehlen. Zwei Beispiele finden sich in Abbildung 130.

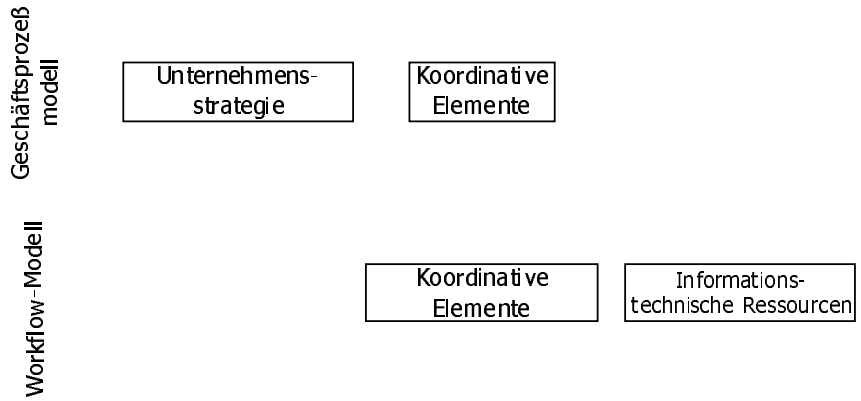


Abbildung 130: Unterschiedliche Informationen in Geschäftsprozeß- und Workflow-Modellen

Ein Realitätsbereich, der typischerweise in Geschäftsprozeßmodellen erfaßt wird, aber nicht in Workflow-Modellen, ist die Unternehmensstrategie. Ein Realitätsbereich, der typischerweise von Workflow- aber nicht von Geschäftsprozeßmodellen erfaßt wird, sind informationstechnische Ressourcen. Aus Realitätsbereichen, die zwar im Workflow- aber nicht im Geschäftsprozeßmodell erfaßt werden, ergibt sich die Anforderung, daß derartige Informationen bei der Bildung des Workflow-Schemas ergänzt werden müssen.

Bereits in Kapitel 2 ist in Form von Aspekten eine Zerlegung für Workflow-Modelle geschaffen worden. Es bietet sich daher an, diese Zerlegung auch auf das Geschäftsprozeßmodell anzuwenden. Durch die Zerlegung von Geschäftsprozeß- und Workflow-Modell mit Hilfe von Aspekten ist es möglich, die Aspekte zu identifizieren, die in beiden Modellen enthalten sind. Ist ein Aspekt im Workflow-Modell enthalten, aber nicht im Geschäftsprozeßmodell, so wird bereits hier die Notwendigkeit für die nachträgliche Ergänzung dieser Informationen erkannt.

Die Anwendung der Aspektseparation wird an Hand des SOM-Verfahrens veranschaulicht. Für SOM kann eine Aspektseparation angegeben werden, die fünf Aspekte umfaßt: Den Kontextaspekt, den Funktionsaspekt, den Verhaltensaspekt, den Informationsaspekt und den Organisationsaspekt. Für das Workflow-Modell braucht sie nicht mehr ausgeführt werden, da dies schon bei der Vorstellung des Zwei-Ebenen-Workflow-Metamodells in Kapitel 6 geschehen ist.

11.4.1.1 Kontextaspekt

Die Aspektseparation des SOM-Modells erfordert die Einführung eines zusätzlichen Aspekts, des Kontextaspekts in Anlehnung an [Kuhn98]. Mit ihm werden besonders die Informationen der obersten Ebene des SOM-Modells erfaßt. Die oberste Ebene des SOM-Modells führt die Modellierung von Unternehmenszielen sowie die Festlegung des Objektsystems des Unternehmens durch. Die Unternehmensziele werden weiter in Sach- und Formalziele unterteilt. Sachziele spezifizieren eine abstrakte Leistung, Formalziele spezifizieren die Art der Leistungserbringung. Ein Ziel kann einem oder mehreren Prozessen zugeordnet sein und jeder Prozeß kann Beiträge zur Erfüllung beliebig vieler Ziele liefern. Aufgabe des Objektsystems ist die Abgrenzung zwischen Diskurswelt und Umwelt. In ihm werden alle Objekte, die mit dem Unternehmen interagieren, aber nicht Gegenstand der Modellierung sind, festgehalten. Beispiele sind Lieferanten, Kunden usw. Der Kontextaspekt wird in Workflow-Schemata nicht direkt wiedergegeben. Dennoch enthält er nützliche Informationen. Sie können besonders bei der Analyse und Modifikation von Workflow-Schemata verwendet werden.

11.4.1.2 Funktionsaspekt

Ausgangspunkt für die Identifizierung des Funktionsaspekts ist die Schachtelung von Haupt- und Serviceprozessen in SOM, die in Abbildung 131 wiedergegeben ist. Die zweite Ebene des SOM-Modells stellt ein Unternehmen als Menge von lose gekoppelten, autonomen Geschäftsprozessen dar. Geschäftsprozesse werden in Haupt- und Serviceprozesse unterteilt. Hauptprozesse dienen der Unterstützung von Unternehmenszielen. Serviceprozesse unterstützen andere Prozesse. Durch die Unterteilung in Haupt- und Serviceprozesse und die unterstützende Funktion von Serviceprozessen entsteht eine funktionale Gliederung mit Serviceprozessen als Unterprozessen.

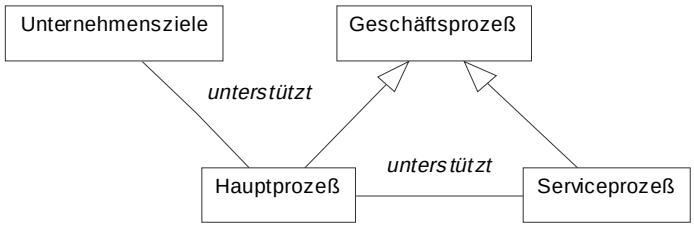


Abbildung 131: Haupt- und Serviceprozesse in SOM

Eine weitere funktionale Zerlegung findet durch die Zerlegung von Geschäftsprozessen in Objekte und Transaktionen statt, sowie die weitere Zerlegung von Objekten und Transaktionen im Rahmen der Objekt- bzw. Transaktionszerlegung. Dies ist in Abbildung 132 dargestellt. Für die Objekt- und Transaktionszerlegung wird durch SOM kein Zerlegungskriterium vorgegeben. Meistens wird die Zerlegung so durchgeführt, daß den Objekten inhaltlich verwandte Aufgabenmengen zugeordnet werden. Auf der Blattebene bestehen Geschäftsprozesse aus einer Menge von Transformationsaufgaben. Daher stellen sie den Endpunkt der funktionalen Zerlegung dar.

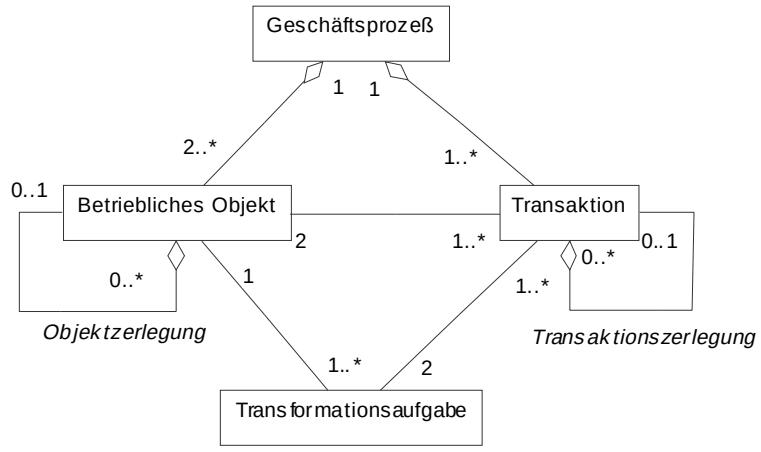


Abbildung 132: Objekt- und Transaktionszerlegungen in SOM

Der Funktionsaspekt wird in SOM durch das Modellelement Prozeß, die Hierarchisierung von Prozessen durch Haupt- und Serviceprozesse und die Zerlegung von Prozessen mit Hilfe von betrieblichen Objekten, Transaktionen und Aufgaben wiedergegeben.

11.4.1.3 Verhaltensaspekt

Die verhaltensorientierte Sicht von SOM beschreibt das Verhalten der Geschäftsprozesse als eine Abfolge von Aufgaben, die über Ereignisse verknüpft sein können, wie in Abbildung 133 dargestellt. Dabei sind große Freiheitsgrade möglich.

Ereignisse werden in Vor- und Nachereignisse differenziert. Ein oder mehrere Vorereignisse stoßen die Ausführung einer Aufgabe an. Ein oder mehrere Nachereignisse zeigen ihre Beendigung an. Vorereignisse können konjunktiv-verknüpft sein, so daß alle Ereignisse eintreten müssen, um die Aufgabe anzustoßen. Sie können aber auch disjunktiv-verknüpft sein, so daß nur ein Ereignis eintreten muß.

Für die Betrachtung der Nachereignisse wird zwischen Entscheidungs- und Transformationsaufgaben unterschieden. Bei Transformationsaufgaben treten immer alle Nachereignisse ein, bei Entscheidungsaufgaben auch eine Teilmenge. Durch Entscheidungsaufgaben und Ereignisse lassen sich also Kontrollflußkonstrukte und auch Schleifen nachbilden. Die Entscheidungsaufgabe verzweigt in Abhängigkeit von der Erfüllung der Bedingung entweder zur Aufgabe am Schleifenanfang oder zur ersten der Schleife folgenden Aufgabe. Der Verhaltensaspekt wird in SOM durch die Konstrukte Ereignis und Entscheidungsaufgabe wiedergegeben.

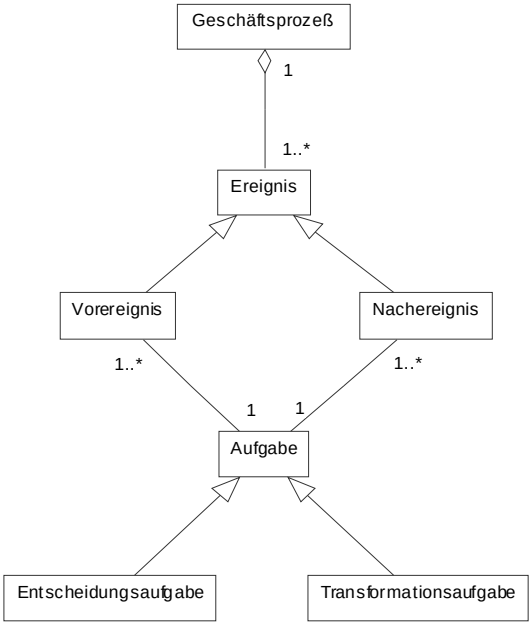


Abbildung 133: Verhaltensorientierte Sicht

11.4.1.4 Informationsaspekt

Der Informationsaspekt wird in SOM nur sehr oberflächlich dargestellt. Es wird davon ausgegangen, daß alle Aufgaben eines betrieblichen Objektes auf einen gemeinsamen Speicher zugreifen. Durch Transaktionen wird außerdem ein Protokoll für den Datenaustausch zwischen betrieblichen Objekten beschrieben. Allerdings können keinerlei Datenstrukturen für den internen Speicher und auch keine konkreten Protokolle für den Datenaustausch angegeben werden. Von Aufgaben benötigte oder produzierte Daten können als Attribute der Aufgabenobjekte angegeben werden. Eine Kopplung der Datenstrukturen zweier Aufgaben kann durch die Attribute der verknüpfenden Ereignisse stattfinden.

11.4.1.5 Organisationsaspekt

Die Darstellung des Organisationsaspekts geschieht in SOM auf der dritten Ebene mit Hilfe der Objekte Organisation, Organisationseinheit, Person und Stelle. Ergebnis ist eine streng hierarchische Struktur, die allerdings keinen weiteren Einschränkungen unterliegt.

11.4.2 Klassifizierungsstrukturen

Mit Hilfe der Aspektseparation konnte ein erstes grobes Raster zur Darstellung der Modelle geschaffen werden. Jetzt gilt es dieses zu verfeinern.

Ein möglicher Ansatz zur feingranularen Darstellung der Modellaspekte ist die Verwendung von Wissensdomänen [Heri98]. Dieses Vorgehen ist zwar für eine statische Abbildung einsetzbar, hat jedoch Nachteile beim Einsatz in einem inkrementellen Abbildungsverfahren. Ursache ist, daß bei Wissensdomänen die Zuordnung eines Schemaelementes zu einem Modellelement nicht an Hand formaler Kriterien, sondern durch manuellen Eingriff des Modellierers erfolgen muß. Ein derartiges Vorgehen ist jedoch für die hier angestrebte inkrementelle Abbildung ungünstig. Vielmehr gilt es die feingranulare Darstellung so zu wählen, daß ein Schemaelement eindeutig klassifiziert und zugeordnet werden kann.

Eine Darstellungstechnik, die dies erlaubt, sind generische und individuelle Konzepte, wie sie in [DoZR97] vorgeschlagen werden. Diese werden in sogenannten Conceptual Supports mit Hilfe von Spezialisierungsbeziehungen streng hierarchisch angeordnet. Durch individuelle

Konzepte werden Objekte wiedergegeben, durch generische Konzepte generische Elemente einer Kategorie von Objekten. Ähnlich wie in der Objektorientierung gilt ein Konzept als Spezialisierung eines anderen, wenn es sämtliche seiner Eigenschaften und darüber hinaus noch weitere besitzt.

Die Modellierung der Aspekte geschieht in Anlehnung an das in [DoZR97] vorgeschlagenen Verfahren mit Hilfe von generischen und individuellen Konzepten. Diese stehen zueinander in „is-a“ Beziehungen und bilden einen azyklischen Graphen mit genau einer Wurzel. Eine Darstellung findet sich in Abbildung 134. Konkrete Beispiele werden im Anschluß bei der Darstellung einzelner Aspekte gegeben. Das Wurzelkonzept stellt dabei ein abstraktes Konzept dar, zu dem kein direktes Äquivalent im Modell existiert. Das Wurzelkonzept wird mit Hilfe generischer Konzepte mit abnehmendem Abstraktionsgrad verfeinert. Dabei finden formale Kriterien, Diskriminatoren genannt, Verwendung. Die Blattebene des gebildeten Graphens besteht aus individuellen Konzepten, die einfachen Modellelementen entsprechen. Für die individuellen Konzepte sind alle diskriminierenden Attribute spezifiziert.

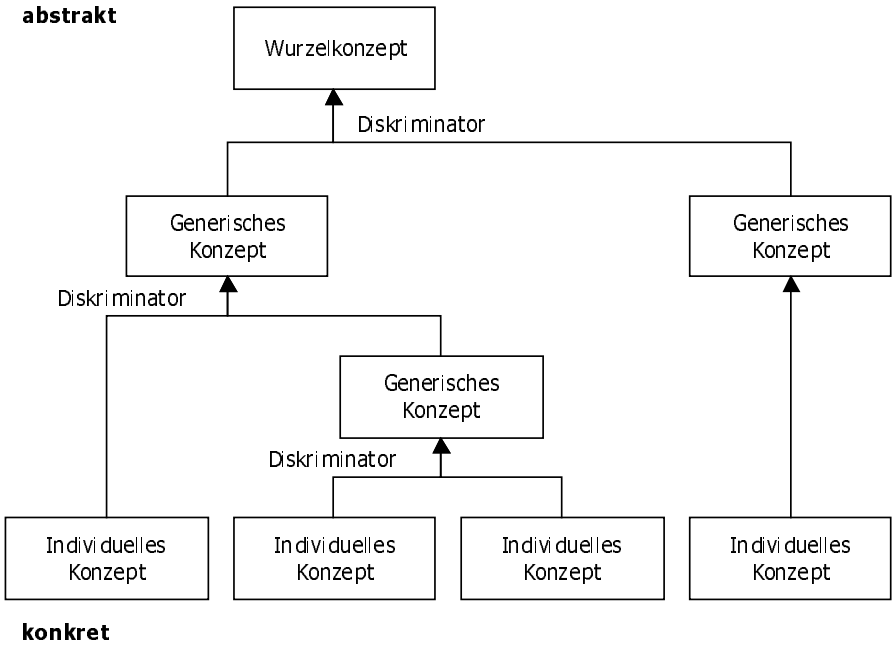


Abbildung 134: Hierarchische Klassifizierungsstruktur

Geschäftsprozeß- und Workflow-Schemata bestehen aus Ausprägungen dieser individuellen Konzepte. Durch Auswertung der diskriminierenden Attribute kann jedes Element eines Schemas exakt einem individuellen Konzept genau eines Aspektes zugeordnet werden.

Im nächsten Schritt werden Klassifizierungsstrukturen in die nach Aspekten getrennte Darstellung des SOM-Modells eingeführt. Hierzu werden die einzelnen Aspektelemente als generische oder individuelle Konzepte dargestellt.

11.4.2.1 Funktionsaspekt

Dem Funktionsaspekt zugehörige Informationen werden in SOM durch die in Abbildung 135 dargestellten Konstrukte Prozeß, Transformationsaufgabe und Betriebliches Objekt wiedergegeben. Sie stammen aus der strukturorientierten Sicht, die in Abbildung 8 dargestellt ist. Die Transformationsaufgaben des SOM-Modells können nach ihrem Automatisierungsgrad unterschieden werden. So kann zwischen vollautomatisierten, teilautomatisierten und nicht-

automatisierbaren Transformationsaufgaben unterschieden werden. Betriebliche Objekte werden in Diskurswelt- und Umweltobjekte unterschieden.

Die Modellkonstrukte, die für das Beispielschema benötigt werden, können jetzt eingeordnet werden. Ergebnis ist die in Abbildung 135 dargestellte Klassifizierungsstruktur für den Funktionsaspekt. Sie enthält die Fertigung, die Auftragsprüfung und die Auftragserfassung als betriebliches Objekt und Kunde als Umweltobjekt. Die Auftragsbearbeitung ist dem Konzept Prozeß zugehörig. „Auftrag erfassen“, Detailprüfung und Standardprüfung sind teilautomatisierte Aufgaben. Das Versenden einer Absage ist hingegen vollautomatisch.

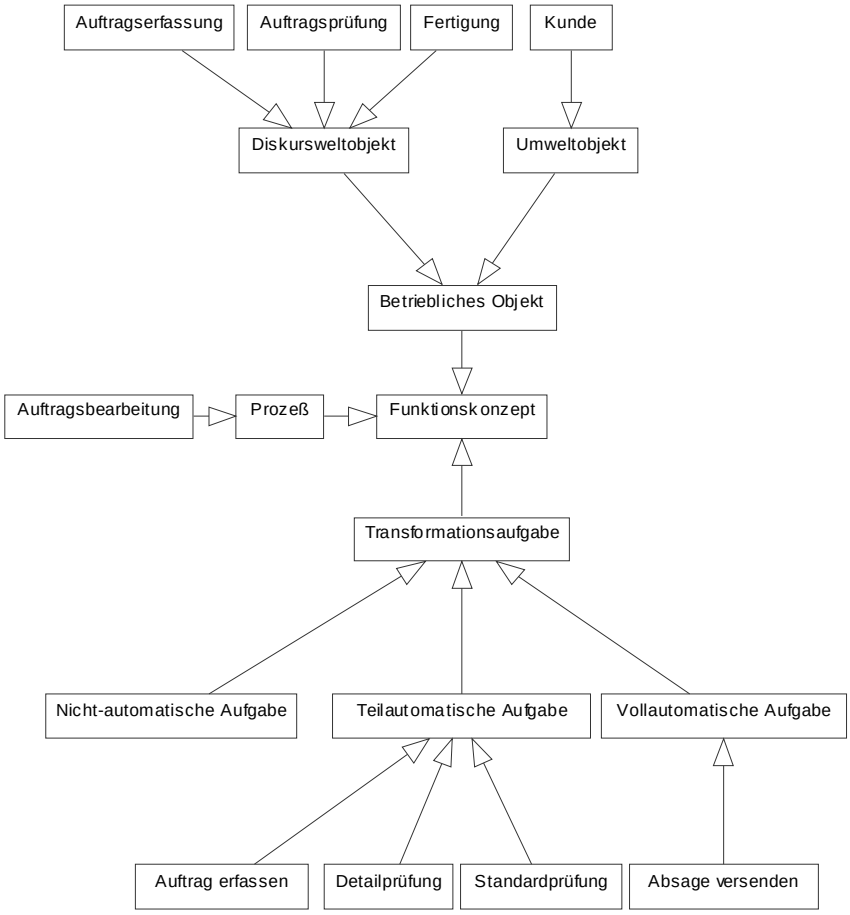


Abbildung 135: Klassifizierungsstrukturen des Funktionsaspekts

11.4.2.2 Verhaltensaspekt

Der Verhaltensaspekt wird durch zwei Bestandteile der in Abbildung 11 dargestellten verhaltensorientierten Sicht repräsentiert. Diese Bestandteile sind Ereignisse und Entscheidungsaufgaben. Ereignisse können in Umwelt- und Transaktions- sowie objektinterne Ereignisse differenziert werden. Die zur Darstellung des Beispielschemas notwendigen Konstrukte sind entsprechend eingeordnet worden und in Abbildung 136 dargestellt.

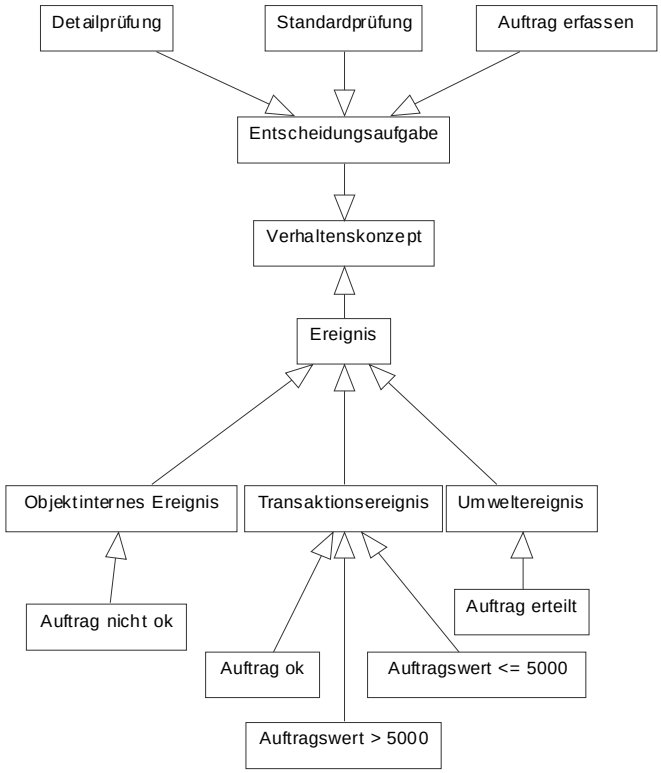


Abbildung 136: Klassifizierungsstrukturen des Verhaltensaspekts

Das SOM-Modell kennt originär keine Schleifenkonstrukte oder bedingte Übergänge. Andererseits kennt das hier entwickelte Workflow-Modell als Kontrollstrukturen Gabelungen und Verknüpfungen sowie Schleifenkonstrukte. Man könnte die Erweiterbarkeit des Workflow-Modells ausnutzen und den SOM-Konstrukten entsprechende Konstrukte in das Workflow-Modell einfügen. Dies würde jedoch Redundanzen heraufbeschwören, die nicht erstrebenswert sein können. Erforderlich ist daher eine Abbildung der semantisch äquivalenten, aber strukturell unterschiedlichen Konstrukte. Dies wird in Abschnitt 11.4.3 behandelt.

11.4.2.3 Organisationsaspekt

Der Organisationsaspekt wird in SOM durch die Elemente Organisation, Stelle, Rolle, Person und organisatorische Einheit verkörpert, wie in Abbildung 137 veranschaulicht. Das SOM-Element Stelle beschreibt eine Funktion innerhalb der Organisation eines Unternehmens. Eine Rolle beschreibt eine Reihe von Fähigkeiten, die eine Person besitzt. Eine Stelle kann immer nur einer Person zugeordnet sein, während eine Rolle mehreren Personen zugewiesen sein kann. Die Konstrukte Sachbearbeiter und Abteilungsleiter werden dem Konzept Rolle zugeordnet. Der Vertrieb ist dem Konzept Organisationseinheit zugehörig.

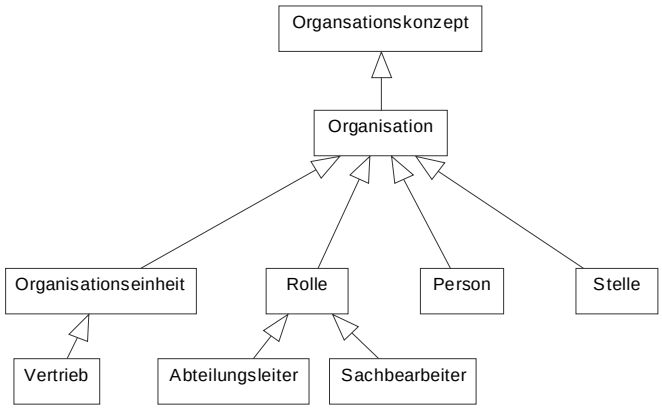


Abbildung 137: Klassifizierungsstrukturen des Organisationsaspekts

11.4.2.4 Informationsaspekt

Wie bereits beschrieben, ist die Darstellung des Informationsaspektes in SOM nur rudimentär. Es wird von einem gemeinsamen Speicher ausgegangen, der allen Aufgaben, die einem betrieblichen Objekt zugehörig sind, zugänglich ist. Weiter wird angenommen, daß betriebliche Transaktionen Protokolle zur Übergabe von Daten zwischen betrieblichen Objekten enthalten. Die Klassifizierungsstrukturen des Informationsaspekts sind in Abbildung 138 dargestellt.

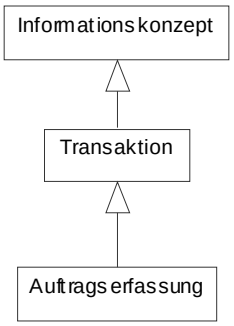


Abbildung 138: Klassifizierungsstrukturen des Informationsaspekts

11.4.3 Äquivalenzsicht

Die Anwendung der oben beschriebenen Klassifizierungsmechanismen lieferte eine Hierarchie von generischen und individuellen Konzepten. Mit ihrer Hilfe lassen sich die Elemente von Geschäftsprozeß- und Workflow-Modell in ein Raster einteilen und gegenüberstellen, um Korrespondenzen ähnlicher Konzepte zu identifizieren. Es kann aber nicht damit gerechnet werden, daß alle semantisch äquivalenten Konstrukte auch strukturell äquivalent sind. Dazu müssen zunächst die möglichen Konstellationen von Konstrukten klassifiziert werden.

Die erste Unterscheidung ist die zwischen elementaren und komplexen Konstrukten, sie ist in Abbildung 139 dargestellt. Elementare Konstrukte bestehen aus einem einzigen Modellelement. Komplexe Konstrukte bestehen aus mehreren Modellelementen, die sich in einer semantisch bedeutungsbehafteten Konstellation befinden. Diese Konstellationen können entweder eindeutig identifiziert werden, oder aber es bedarf zusätzlicher Informationen bzw. der Interpretation durch den Modellierer. Im ersten Fall spricht man von konstituierenden Konstrukten, im zweiten Fall von fakultativen Konstrukten.

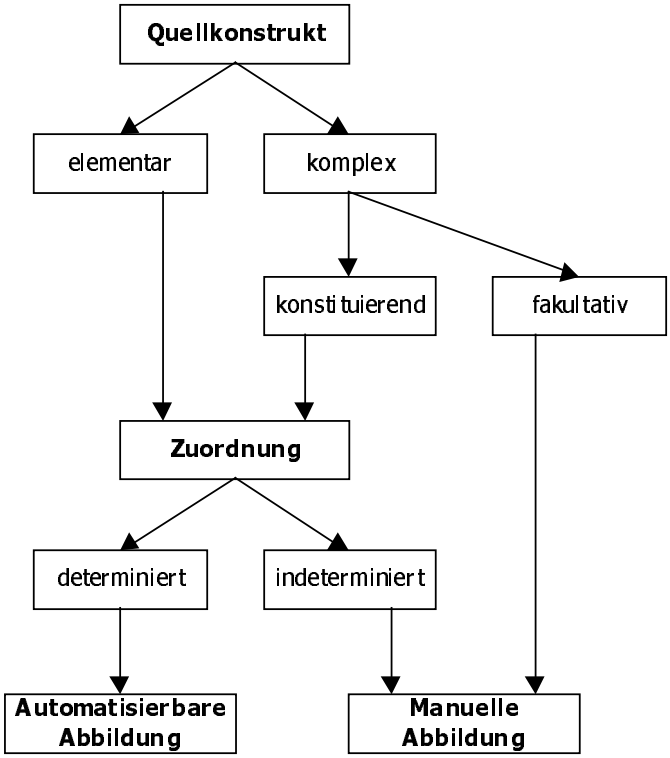


Abbildung 139: Konstrukte und ihre Abbildung

Die Verknüpfung von Konstrukten des Geschäftsprozeßmodells zu denen des Workflow-Modells kann determiniert oder indeterminiert sein. Determiniert bedeutet, daß es zum Konstrukt des Geschäftsprozeßmodells genau ein Konstrukt des Workflow-Modells gibt. Indeterminiert bedeutet, daß mehrere äquivalente Konstrukte existieren, deren Auswahl durch den Modellierer geschehen muß, da keine expliziten oder extrahierbaren impliziten Unterscheidungsmerkmale vorhanden sind. Bei einer determinierten Verknüpfung kann eine automatisierbare Abbildung erfolgen, bei einer indeterminierten bzw. einer fakultativ komplexen Verknüpfung nur eine manuelle.

Um die strukturelle Äquivalenz zwischen dem Geschäftsprozeß- und dem Workflow-Modell herzustellen, müssen Abbildungen zwischen elementaren oder konstituierenden komplexen Konstrukten des Geschäftsprozeßmodells und semantisch äquivalenten einfachen oder komplexen Konstrukten des Workflow-Modells identifiziert werden. Die im vorangegangenen Abschnitt entwickelten Konzepte behandeln nur einfache Konstrukte. Sie enthalten keine Mechanismen für die Erfassung komplexer Konstrukte. Es muß daher eine Möglichkeit geschaffen werden, komplexe Modellkonstrukte zugänglich zu machen. Ähnliche Probleme finden sich bei der Sichtenkonsolidierung im Datenbankbereich wieder, wie beispielsweise in [LaLo95] beschrieben. Eine aus diesem Bereich stammende geeignete Lösung ist die informationserhaltende Schematransformation. Als ungelöstes Problem verbleibt dabei jedoch die Abbildung komplexer Konstrukte. Dies geschieht mit Hilfe eines Topologieerkennungsmechanismusses. Um die in konstituierenden komplexen Konstrukten enthaltenen Zusatzinformationen ausdrücken zu können, werden in die Äquivalenzsicht Konzepte eingebracht, die die Semantik konstituierender komplexer Konstrukte wiedergeben. Ihnen wird eine Topologie zugeordnet, die eine eindeutige Identifizierung der Konstrukte im Schema ermöglicht. Auf diese Weise können Algorithmen entwickelt werden, die ein Schema automatisch auf das Vorhandensein bestimmter Topologien überprüfen.

Dies wird am Beispiel des Verhaltensaspekts erläutert. Das SOM-Modell kennt originär keine Schleifenkonstrukte oder bedingte Übergänge. Andererseits kennt das hier entwickelte Workflow-Modell als Kontrollstrukturen nur Gabelungen und Verknüpfungen sowie Schleifenkonstrukte. Mit den oben vorgestellten Techniken lassen sich die Kontrollstrukturen des SOM-Modells in Konstrukte umwandeln, die strukturell zu denen des Workflow-Modells kompatibel sind. So kann das Konstrukt bedingte Verzweigung auf der Basis des in Abbildung 140 dargestellten komplexen Konstrukts identifiziert werden. Auch das Konstrukt Schleife kann durch die Topologieerkennung aus den ereignisorientierten Basiskonstrukten von SOM umgewandelt werden.

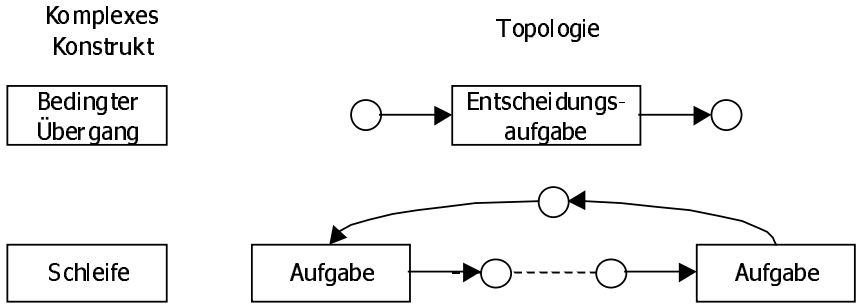


Abbildung 140: Topologien der komplexen Konstrukte

Die durch Topologieerkennung auffindbaren konstitutiven komplexen Konstrukte erweitern die Klassifizierungsstrukturen zur Äquivalenzsicht. Für den Verhaltensaspekt ist diese in Abbildung 141 dargestellt.

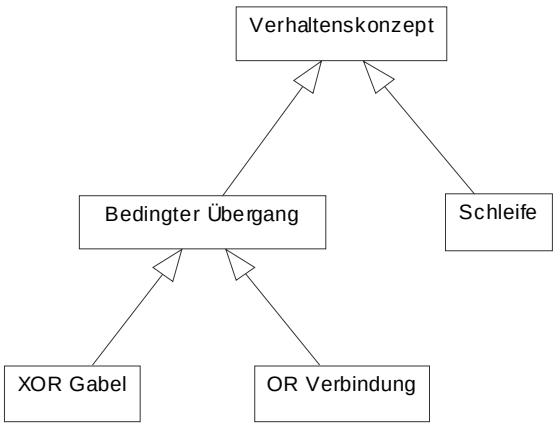


Abbildung 141: Äquivalenzsicht für den Verhaltensaspekt

Die Topologieerkennung ist auch ein Mittel, um eine Menge von Konstrukten abzubilden, die zwar jedes für sich abbildbar sind, jedoch in einer Konstellation stehen, die im Zielschema unzulässig ist. Ein Beispiel ist die Zyklenfreiheit, die von vielen Workflow-Modellen von Schemata gefordert wird, aber von Geschäftsprozeßmodellen durchaus vorgesehen ist.

11.4.4 Abbildungsoperatoren

Mit den vorangegangenen Konzepten der Aspektseparation, Klassifizierungsstrukturen und Äquivalenzsicht konnte ein geeignetes Raster für die Darstellung von Geschäftsprozeß- und Workflow-Modell erreicht werden. Daher kann der zweite identifizierte Problembereiche behandelt werden, die Schaffung geeigneter Verknüpfungen zwischen Geschäftsprozeß- und

Workflow-Modell. Auf ihrer Grundlage werden dann Operatoren geschaffen, die eine inkrementelle Abbildung ermöglichen. Dargestellt ist dies in Abbildung 142.

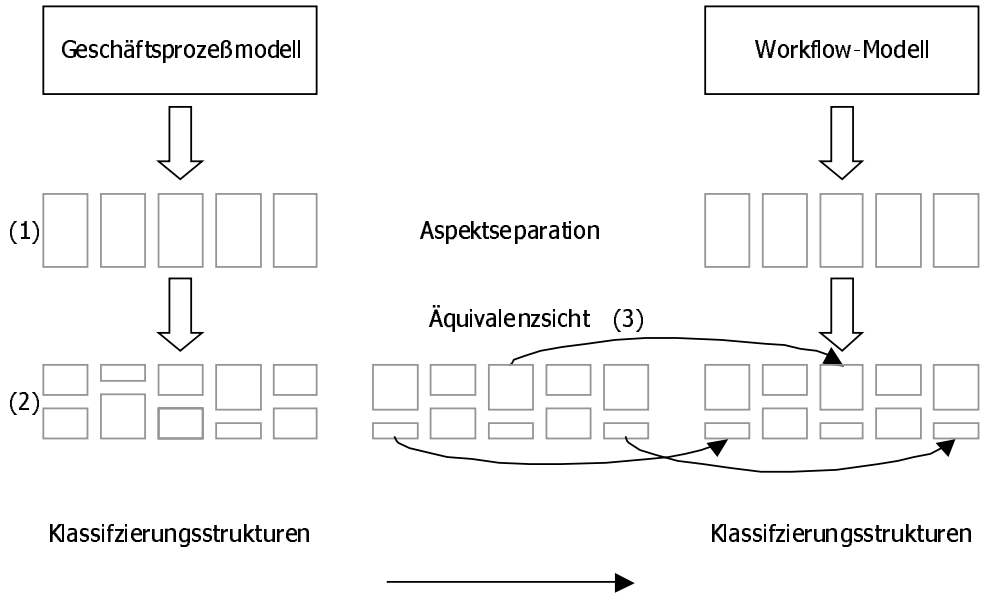


Abbildung 142: Bestimmung von Abbildungsoperatoren

Ausgangspunkt für die Schaffung von Abbildungsoperatoren ist die auf der Basis des Geschäftsprozeßmodells gebildete Äquivalenzsicht und das aspektseparierte und feinstrukturierte Workflow-Modell. Zur Bildung der Abbildungsoperatoren werden Aspekt für Aspekt die Äquivalenzsicht und das Workflow-Modell verglichen und Abbildungsbeziehungen identifiziert. Das Vorgehen beginnt für jeden Aspekt jeweils mit dem Wurzelelement in der Äquivalenzsicht. Von diesem ausgehend wird der Klassifizierungsgraph durchlaufen und für alle Knoten geprüft, ob im Workflow-Modell Konzepte vorhanden sind, zu denen ein Abbildungsoperator definiert werden kann. Der günstigste Fall ist natürlich, wenn die Abbildungsbeziehung zwischen Wurzelknoten der Klassifizierungsgraphen stattfinden kann, bedeutet dies doch die Möglichkeit einer automatisierbaren Abbildung.

Kann für ein Konzept des Geschäftsprozeßmodells kein passendes Konzept im Workflow-Modell gefunden werden, kommen die besonderen Eigenschaften des in dieser Arbeit entwickelten Ansatzes zum Tragen. Durch die Erweiterbarkeit des Workflow-Modells hat man die Möglichkeit zu wählen, ob man entweder ein zusätzliches Element des Workflow-Modells schafft, oder aber eine Abbildung auf mehrere bereits vorhandene Konzepte durchführt, wie dies bei den bisherigen Ansätzen erforderlich ist. Die Bildung von Abbildungsregeln wird an Hand der Elemente des Verhaltensaspekts erläutert. In Abbildung 143 sind dazu die Klassifizierungsstrukturen des Verhaltensaspekts für ein Geschäftsprozeß- und ein Workflow-Modell dargestellt. Zwischen diesen sind Verbindungslinien zur Wiedergabe der Abbildungsbeziehungen eingetragen. Dabei wird auch der Nutzen der vorher gebildeten Äquivalenzsicht deutlich. Die durch die Topologieerkennung identifizierten konstitutiven komplexen Konstrukte XOR-Gabel und OR-Verbindung können direkt in entsprechende Konstrukte des Workflow-Modells umgesetzt werden.

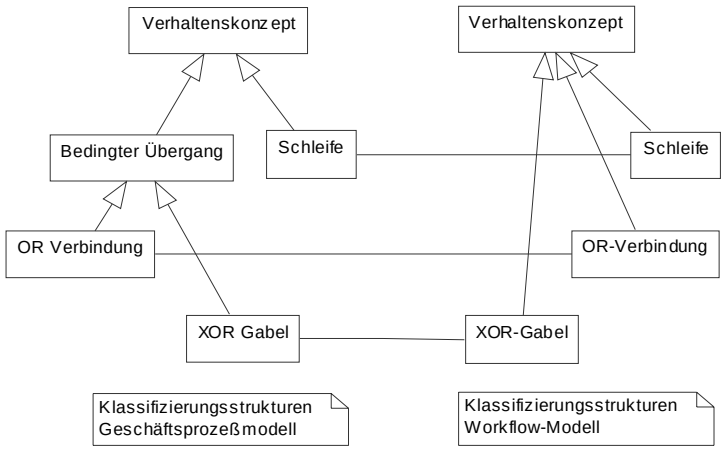


Abbildung 143: Abbildungsregeln für den Verhaltensaspekt

11.5 Die Durchführung der Abbildung

Durch die Bestimmung eines genügend feinen Rasters für die Darstellung von Geschäftsprozeß- und Workflow-Schema und die Definition von Abbildungsoperatoren wurde die Grundlage für die Durchführung von Abbildungen gelegt. Die hierfür durchgeführten Schritte sind in Abbildung 144 angeordnet, und durch Nummern in der Abbildung kenntlich gemacht. Es handelt sich um die Aspektseparation (1), Klassifizierungsstrukturen (2), die Äquivalenzsicht (3) und Abbildungsoperatoren (4).

Mit ihrer Hilfe lassen sich automatisierbare Abbildungen zwischen Quell- und Zielschema herstellen. Auf der Schemaebene arbeiten diese Kernkonzepte wie folgt zusammen, wobei die Nummern (5-7) in Abbildung 144 eine Orientierung geben.

5. Das Geschäftsprozeßschema wird mit Hilfe der Äquivalenzsicht in ein semantisch äquivalentes Schema umgewandelt.

6. Unter Nutzung der Abbildungsoperatoren wird dieses äquivalente Schema auf ein Workflow-Schema-Gerüst abgebildet.

7. Dieses Workflow-Gerüst wird um notwendige Angaben ergänzt, die im Geschäftsprozeßschema fehlen, aber für ein vollständiges Workflow-Schema notwendig sind.

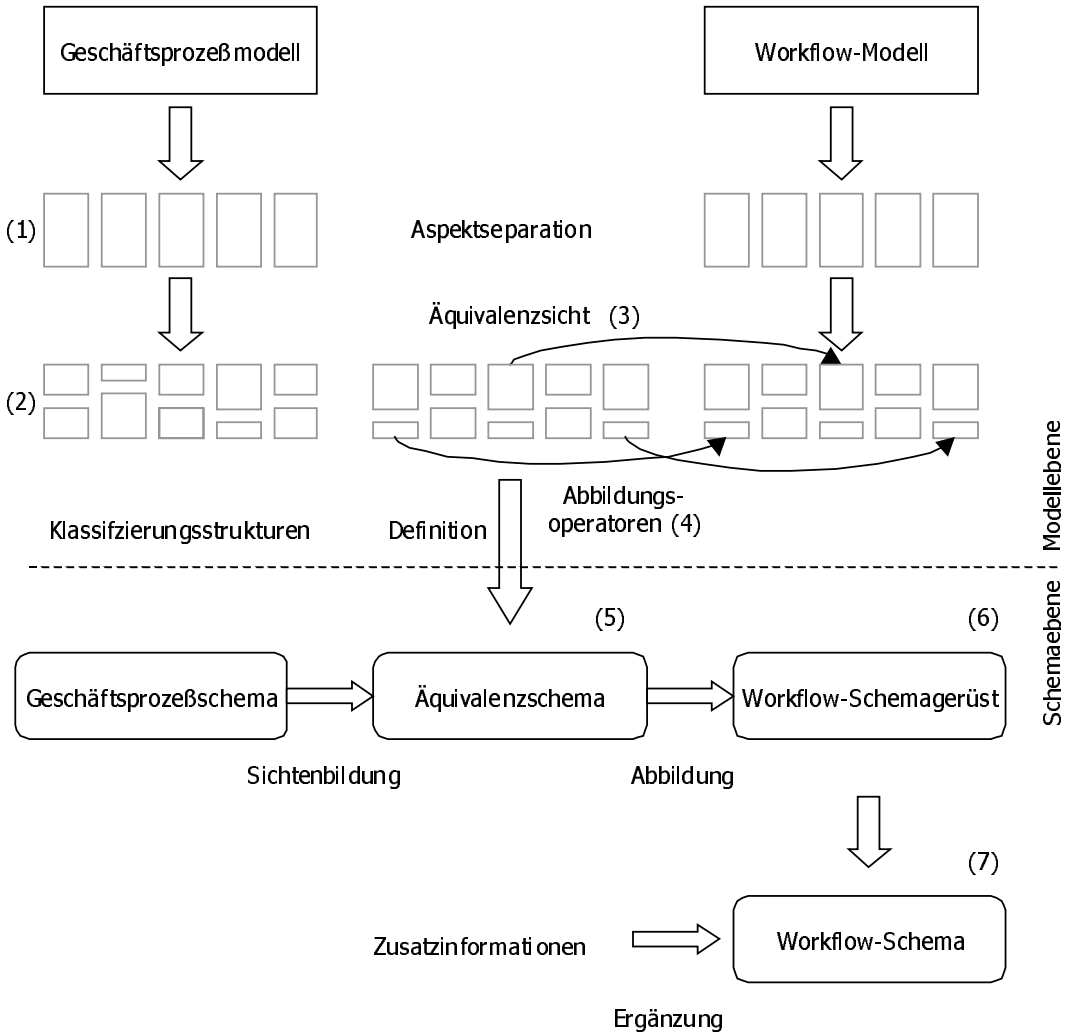


Abbildung 144: Konzeption des Abbildungsverfahrens

11.5.1 Äquivalenzschema

Zur Abbildung des Beispielschemas in Abbildung 127 wird zunächst ein Äquivalenzschema auf der Basis der in Abschnitt 11.4.3 entwickelten Äquivalenzsicht geschaffen. Dazu muß insbesondere eine Topologieerkennung zum Auffinden von konstitutiven komplexen Konstrukten durchgeführt werden. Die Topologieerkennung identifiziert drei konstitutiv komplexe Konstrukte in Form der Elemente XOR-Gabel und OR-Verbindung. Bei den XOR-Gabel-Konstrukten ist jeweils das Entscheidungskriterium für die Gabelung angegeben. Das sich daraus ergebende Äquivalenzschema ist in Abbildung 145 dargestellt.

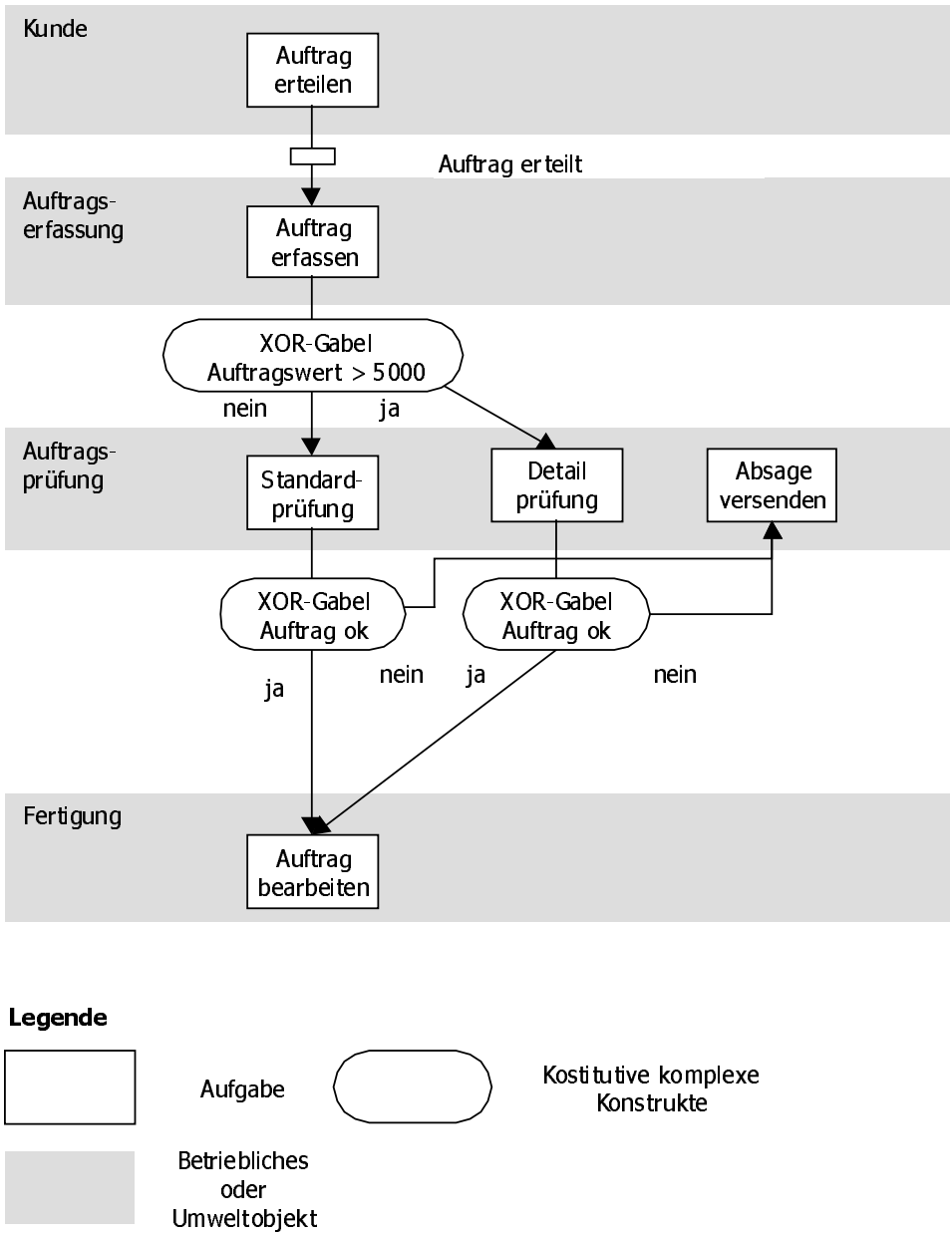


Abbildung 145: Äquivalenzschema

11.5.2 Workflow-Schema-Gerüst

Durch die Anwendung der Abbildungsregeln kann ein Workflow-Schema-Gerüst gebildet werden, das in Abbildung 146 dargestellt ist. Es enthält Schemaelemente, die auf der Basis des Workflow-Modells gebildet wurden und durch die Anwendung der Abbildungsregeln aus dem Geschäftsprozeßschema abgeleitet wurden.

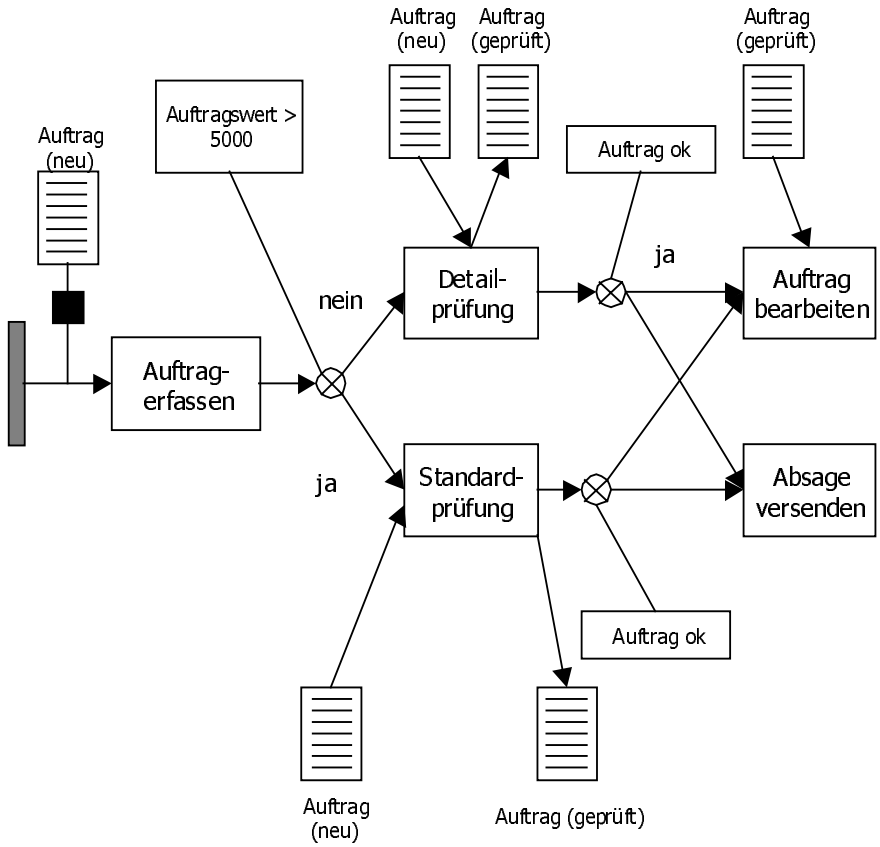


Abbildung 146: Beispiel-Workflow-Schema

11.5.3 Workflow-Schema

Der letzte Schritt ist die Ergänzung eventueller Lücken im Workflow-Schema-Gerüst durch den Modellierer. Diese können entstehen, wenn beispielsweise zu wenige Informationen im Geschäftsprozeßschema enthalten waren. Da dies in diesem Beispiel jedoch nicht der Fall ist, entspricht das endgültige Workflow-Schema dem Workflow-Schema-Gerüst. Auf eine nochmalige Darstellung kann daher verzichtet werden.

11.6 Bewertung

Jetzt wird untersucht, ob die angestrebte inkrementelle Abbildung mit Hilfe des hier entwickelten Verfahrens erreicht werden kann. Ausgangspunkt ist eine Erweiterung des Unternehmensplans, der nicht mehr nur die Gewinnmaximierung, sondern auch die Verbesserung des öffentlichen Erscheinungsbildes als Unternehmensziel ansieht. Dies schlägt sich in einer Reihe von Erweiterungen auf den Ebenen Unternehmensplan, Geschäftsprozesse und Ressourcen nieder.

Der Unternehmensplan wird um das Ziel der Verbesserung des öffentlichen Erscheinungsbildes erweitert. Eine wichtige Strategie soll hierzu sein, bei den Geschäftsbeziehungen die Konformität mit außenwirtschaftlichen Bestimmungen einzuhalten. Es soll angenommen werden, daß das Unternehmen in der Vergangenheit durch die mangelnde Einhaltung dieser Bestimmungen im öffentlichen Meinungsbild Schaden genommen hat. Der dementsprechend erweiterte Unternehmensplan ist in Abbildung 147 dargestellt.

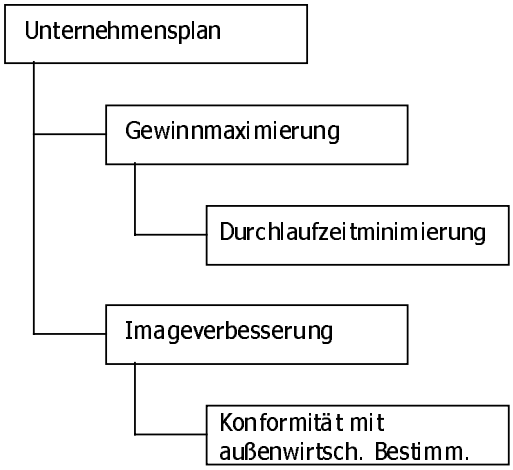


Abbildung 147: Erweiterter Unternehmensplan (Ausschnitt)

Die oberste Ebene der Geschäftsprozesse wird durch die Strategieerweiterung noch nicht betroffen. Erst die Detaillierung des in Abbildung 125 wiedergegebenen Geschäftsprozesses zeigt Änderungen auf. So enthält die Detaillierung des betrieblichen Objekts Vertrieb ein zusätzliches betriebliches Objekt Embargoprüfung. Dies ist in Abbildung 148 dargestellt.

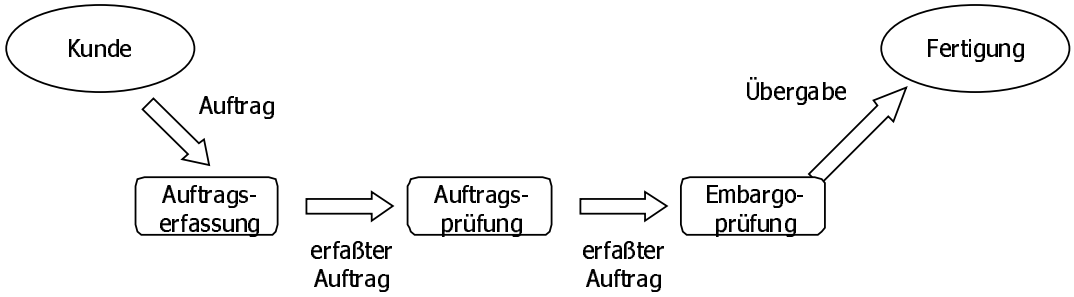


Abbildung 148: Erweiterter Geschäftsprozeß

In Abbildung 149 ist ein um die Embargoprüfung erweitertes Vorgangs-Ereignis-Schema dargestellt. Bei einer nicht bestandenen Embargoprüfung soll eine Absage versandt werden, sonst soll die Bearbeitung des Auftrags wie bisher weitergeführt werden.

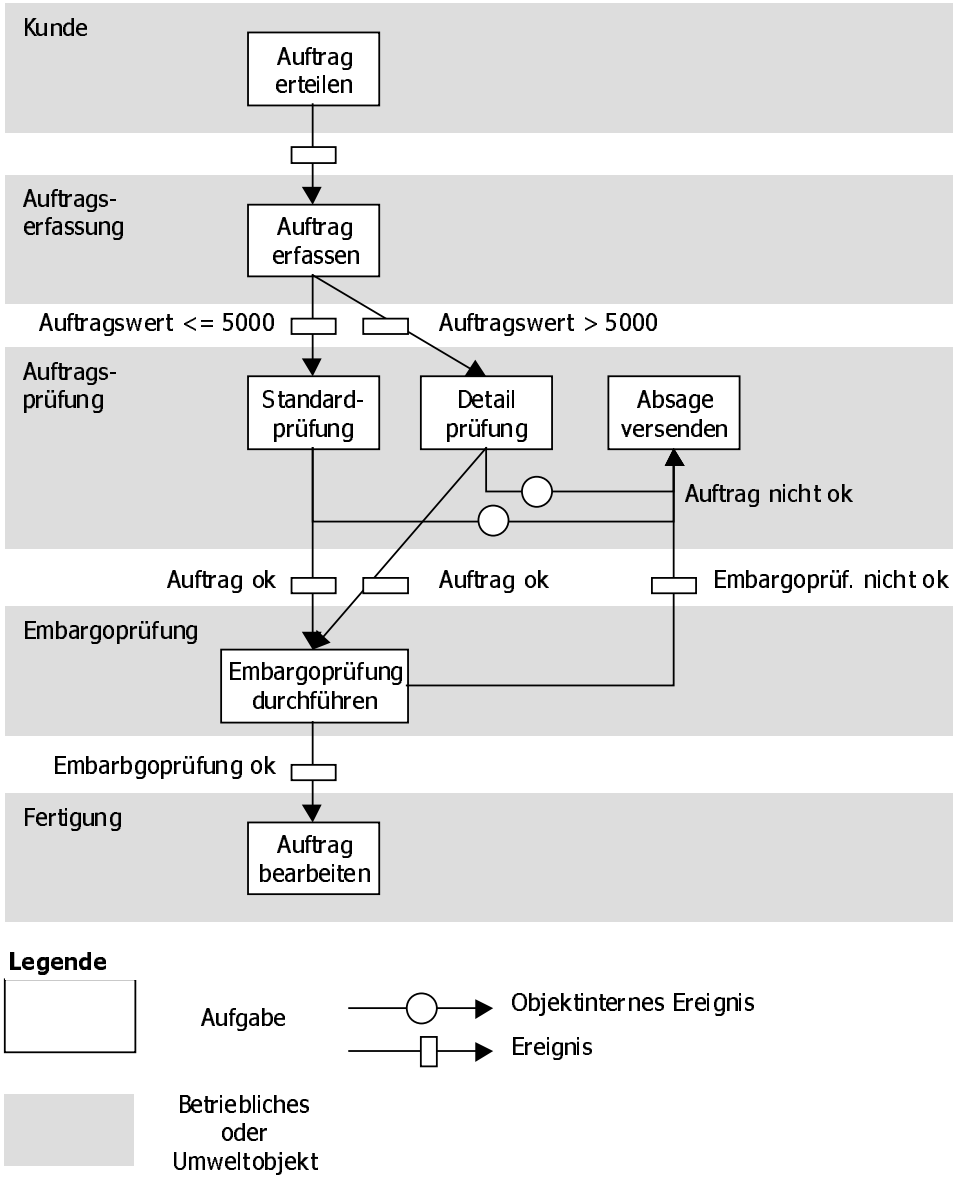


Abbildung 149: Vorgangs-Ereignis-Schema

Es wird angenommen, daß die für die Abbildung des Schemas notwendigen Schritte auf Modellebene bereits früher für ein Schema durchgeführt wurden und daher nicht mehr dargestellt werden brauche. Ausgehend von dem erweiterten Vorgangs-Ereignis-Schema kann dann ein erweitertes Äquivalenzschema geschaffen werden, das in Abbildung 150 dargestellt ist. Das Raster, das durch die Anwendung von Aspektseparation, Klassifizierungsstrukturen und Äquivalenzschema erzeugt wird, ist durch gestrichelte Linien wiedergegeben.

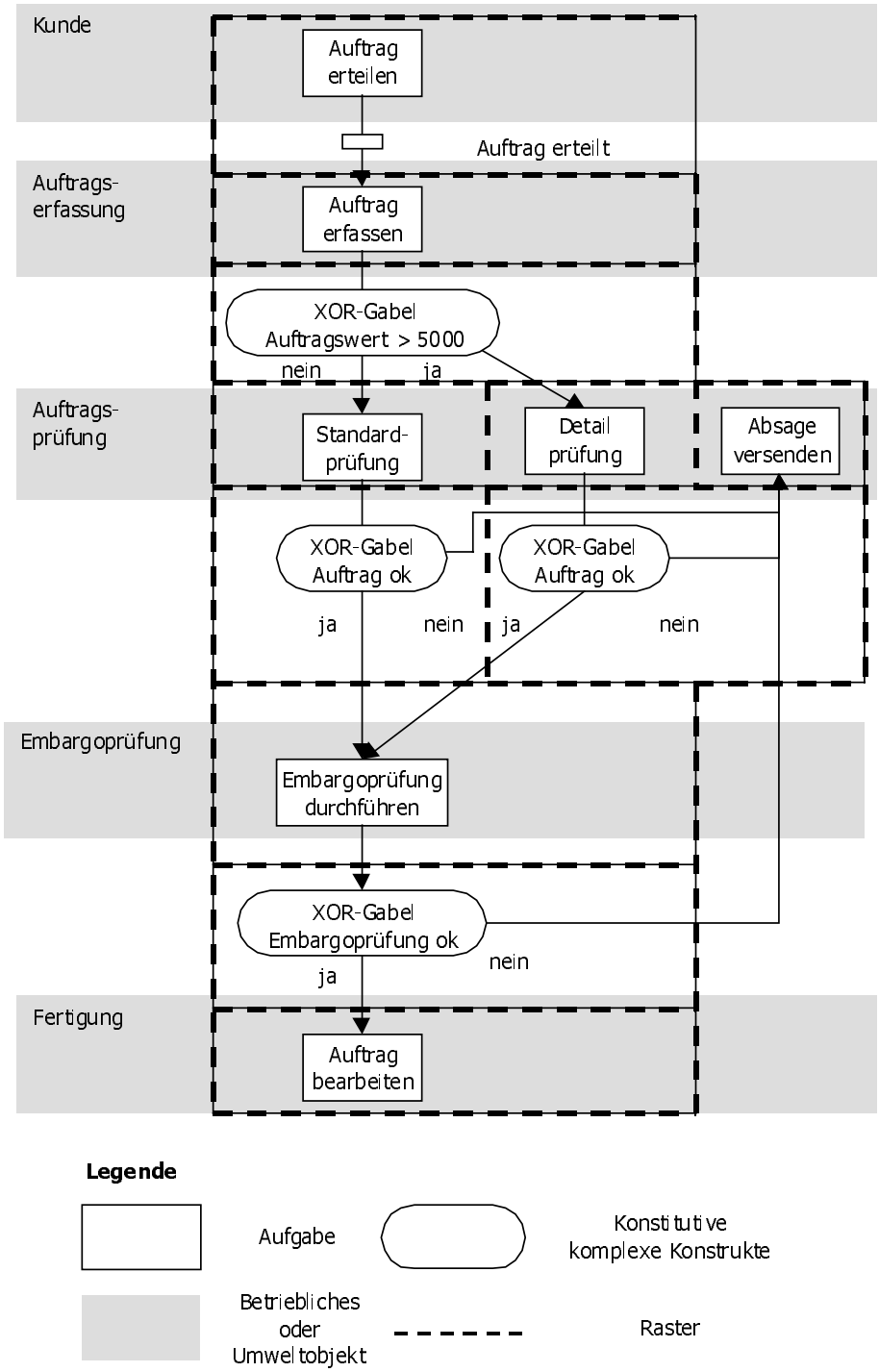


Abbildung 150: Erweitertes Äquivalenzschema

Durch die Rasterung können die Elemente des Schemas erfaßt werden, die verändert worden sind. Da die Abbildungsregeln auf der Ebene individueller Konstrukte definiert sind, brauchen für die nun notwendige erneute Abbildung nur die von der Änderung betroffenen Elemente verändert werden. Das dabei erhaltene erweiterte Workflow-Schema ist in Abbildung 151 dargestellt.

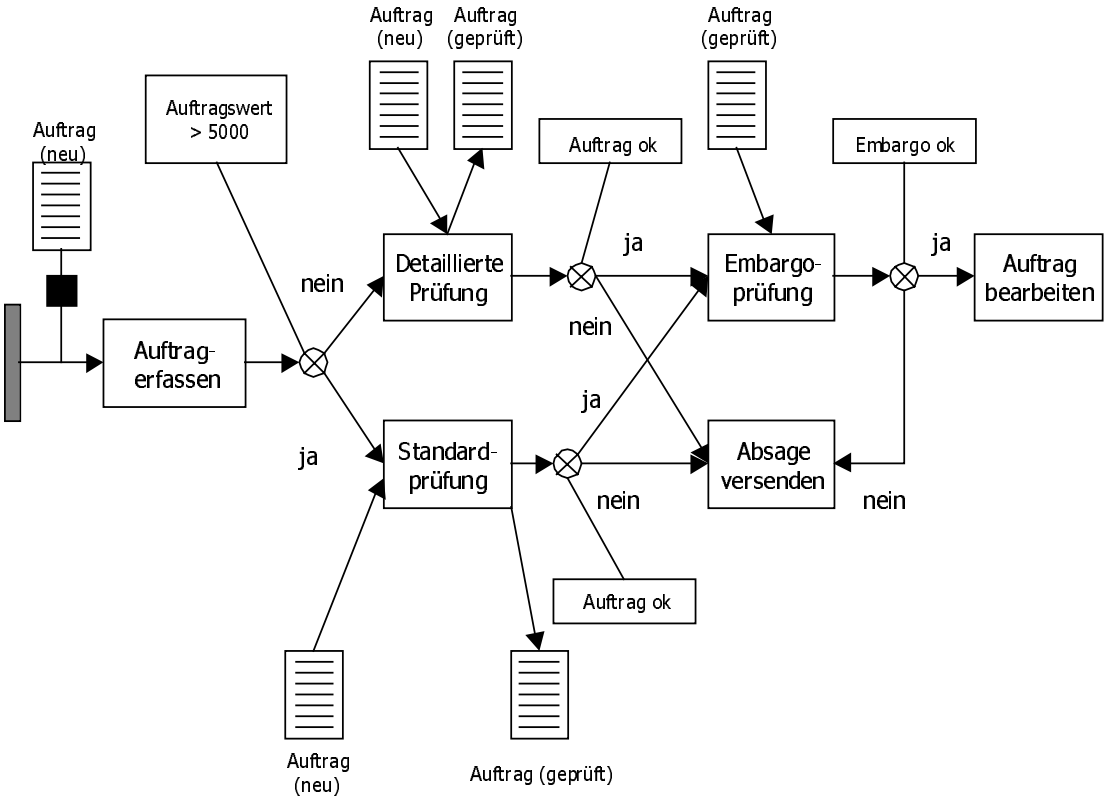


Abbildung 151: Erweitertes Workflow-Schema

Als Fazit ist zu ziehen, daß die angestrebte inkrementelle Abbildung erreicht wird. Durch das entwickelte Raster werden feingranulare Änderungssphären geschaffen. Durch sie können die von einer Änderung betroffenen Elemente eingrenzt werden. Damit läßt sich eine inkrementelle Abbildung durch Anwendung der Abbildungsoperatoren erreichen.

11.7 Zusammenfassung

In diesem Kapitel wurde das Ziel der Schaffung eines inkrementellen Abbildungsverfahrens für Geschäftsprozeß- auf Workflow-Schemata erreicht. Hierfür wurde ein Raster für die Darstellung von Geschäftsprozeß- und Workflow-Modellen geschaffen, das als Grundlage für Abbildungsoperatoren einer inkrementellen Abbildung dient.

Ebenfalls ein Fortschritt stellt die Einbeziehung aller vom SOM-Modell bereitgestellten Informationen in den Abbildungsprozeß dar. Existierende Verfahren verwendeten nur einen Ausschnitt der zur Verfügung stehenden Informationen. Durch den fein gegliederten Abbildungsprozeß konnte eine saubere methodische Fundierung erreicht werden, ein Punkt, in dem existierende Ansätze ebenfalls Schwächen aufweisen.

Um das Raster zur Darstellung von Geschäftsprozeß- und Workflow-Modell zu schaffen, wurden drei Konzepte entwickelt. Es sind dies die Aspektseparation, Klassifizierungsstrukturen und die Schaffung einer Äquivalenzsicht. Mit Hilfe der Aspektseparation konnte ein erstes, grobes Raster geschaffen werden, das die Darstellung des Informationsgehalts von Geschäftsprozeß- und Workflow-Modell erlaubt. Durch Klassifizierungsstrukturen wurde dann eine weitere Zerlegung der einzelnen Aspekte durchgeführt. Die Äquivalenzsicht legte die Grundlage für die Definition von Verknüpfungen zwischen semantisch äquivalenten, aber strukturell verschiedenen Konstrukten in Geschäftsprozeß- und Workflow-Modell. Auf der Basis des so geschaffenen Rasters konnten dann Abbildungsoperatoren definiert werden.

Die auf diese Weise geschaffenen Abbildungsoperatoren erlauben eine inkrementelle Abbildung. Dazu wird das Geschäftsprozeßschema mit Hilfe der Äquivalenzsicht in ein semantisch äquivalentes Schema umgewandelt. Durch Anwendung der Abbildungsoperatoren wird dieses äquivalente Schema auf ein Workflow-Schema-Gerüst abgebildet. Schließlich wird das Workflow-Schema-Gerüst um Informationen ergänzt, die im Geschäftsprozeßschema fehlen, aber für ein vollständiges Workflow-Schema notwendig sind.

12 Implementierung

Dieses Kapitel und das darauffolgende Kapitel 13 stehen in einem engen Zusammenhang. In diesem Kapitel wird der Einsatz der in dieser Arbeit entwickelten Konzepte im Rahmen mehrerer Projekte dargestellt. Die dabei gemachten Erfahrungen werden im Kapitel 13 aufgegriffen, um auf ihrer Grundlage die Evaluierung der Konzepte durchzuführen.

Die Konzepte dieser Arbeit wurden mehrfach in Projekten eingesetzt. In Abbildung 152 ist dargestellt, welche Konzepte in welchen Projekten implementiert und verfeinert wurden. Im Projekt „Workflow-orientierte Geschäftsprozeßimplementierung auf der Basis von Lotus Notes“ [Weich97], kurz WOGE, wurde die aspektelementorientierte Schemarepräsentation eingesetzt. Grundlage für die Gestaltung des Workflow-Modells im WOGE-Projekt war das Zwei-Ebenen-Workflow-Modell. Im Rahmen dieses Projektes konnten zudem zwei Anforderungen identifiziert werden. So erwies sich die in WOGE eingesetzte Gesamtabbildung des Geschäftsprozeßschemas bei häufigen Änderungen als zu aufwendig. Hieraus ergab sich die Notwendigkeit eines inkrementellen Abbildungsverfahrens. Darüber hinaus wurde auch deutlich, daß Veränderungen der Geschäftsprozeßschemata auftauchen können, die zur informationstechnischen Umsetzung eine Erweiterung des Workflow-Modells notwendig machen. Im CompFlow-Projekt (Component-based Workflow support) [ScAs98b] wurde daher ein inkrementelles Abbildungsverfahren für Geschäftsprozeßschemata und eine erweiterbare Repräsentation des Workflow-Modells geschaffen. Außerdem wurde im CompFlow-Projekt das Konzept der aspektelementorientierten Schemarepräsentation weiterentwickelt. Das Zwei-Ebenen-Workflow-Metamodell bildete ebenfalls eine wichtige Grundlage. Das im CompFlow-Projekt entwickelte inkrementelle Abbildungsverfahren wurde im SOMFlow-Projekt [Scho98] eingesetzt und soweit verfeinert, daß sogar bijektive Abbildungsoperatoren zwischen Geschäftsprozeß- und Workflow-Schemata geschaffen werden konnten, die eine konzertierte Evolution beider Schemata erlauben. Sämtliche Konzepte dieser Arbeit werden im VORREITER-Projekt verwendet. Dieses befindet sich zwar noch in einer frühen Phase, dennoch lassen sich bereits erste Erfahrungen festhalten.

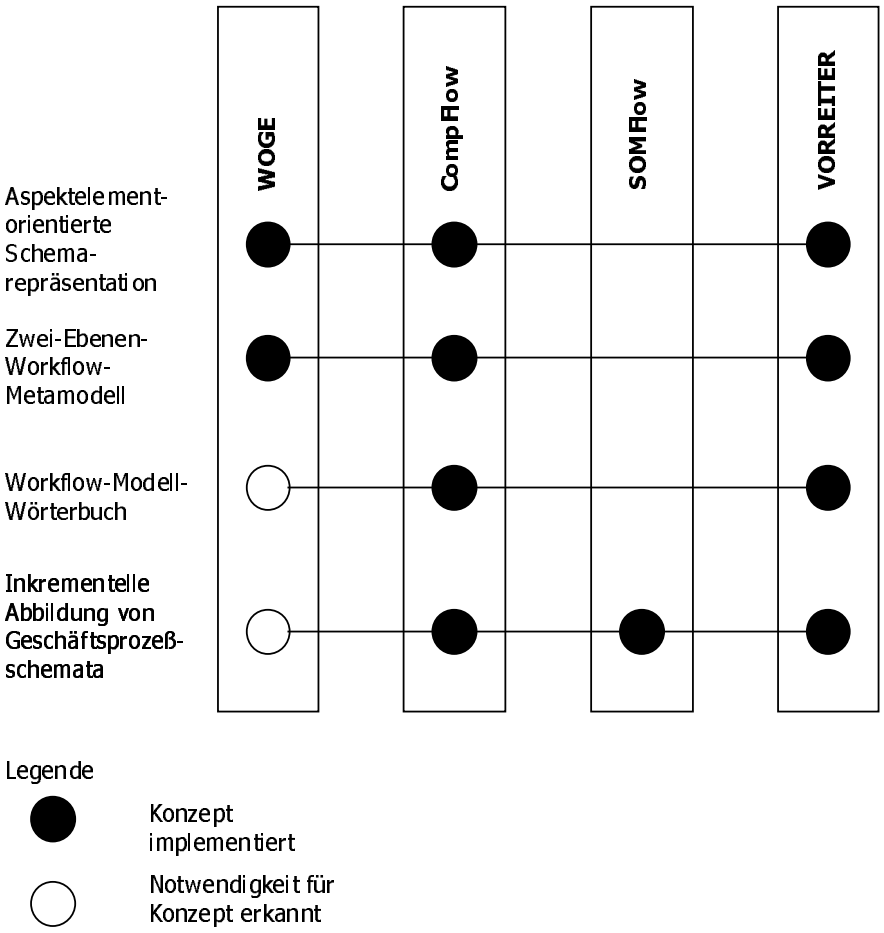


Abbildung 152: Einsatz der Konzepte in Projekten

Die Darstellung der Projekte wird nach folgendem Muster durchgeführt: Für jedes Projekt werden zunächst das Projektziel, sowie die einzuhaltenden Rahmenbedingungen dargestellt. Danach wird der verfolgte Lösungsweg beschrieben. Den Abschluß bildet jeweils die Bewertung der Projektergebnisse unter Maßgabe der Projektziele.

12.1 WOGE

12.1.1 Projektziel

Ziel des Projekts „Workflow-orientierte Geschäftsprozeßimplementierung auf der Basis von Lotus Notes“ [Weich97], kurz WOGE, war es, den mit SOM modellierten Geschäftsprozeß „Client-Server-Projektangebot“, kurz CSPA, in Lotus Notes [Note] umzusetzen. Dabei waren zwei Anforderungen zu erfüllen. Die erste war, daß Anpassungen der informationstechnischen Umsetzung an Prozeßänderungen flexibel durchführbar sein mußten. Die zweite Anforderung war, daß dabei die Autarkie bei der Ressourcenzuordnung berücksichtigt wird. So sollte kein zusätzliches Produkt erforderlich sein, um die Prozeßunterstützung zu bewerkstelligen. Statt dessen sollten nur die von Lotus Notes bereitgestellten Mechanismen verwendet werden. Für die Abbildung von Geschäftsprozeß- auf Workflow-Schemata wurde ein manuelles, informationstechnisch nicht unterstütztes Abbildungsverfahren für ausreichend gehalten. Ebenfalls wurde davon ausgegangen, daß die Menge benötigten Workflow-Schemaelemente a priori bestimmbar ist. Das Workflow-Modell wurde daher statisch repräsentiert.

12.1.2 Lösungsansatz

Um die Anforderungen des WOGE-Projekts zu erfüllen, wurden das Zwei-Ebenen-Workflow-Metamodell und die aspektelementorientierte Schemarepräsentation eingesetzt.

Auf Basis des Zwei-Ebenen-Workflow-Metamodells wurde zunächst ein Workflow-Modell gebildet. Der in der SOM-Notation dargestellte Geschäftsprozeß CSPA wurde dann auf die Workflow-Schema abgebildet, dessen Elemente durch das gebildete Workflow-Modell bereitgestellt werden. Für dieses Workflow-Schema mußte dann eine anpaßbare Schemarepräsentation geschaffen werden. Dazu wurde eine aspektelementorientierte Schemarepräsentation gebildet, die mit Hilfe einer kompositen Anwendung implementiert wurde.

Das Problem lag dabei darin, daß Lotus Notes von sich aus kein komponentenorientiertes Rahmenwerk darstellt. Daher mußten die Bestandteile von Lotus Notes so angeordnet werden, daß die in dieser Arbeit entwickelten Konzepte auf sie angewendet werden konnten. Die Ergebnisse dieser Überlegungen sind in Abbildung 153 veranschaulicht.

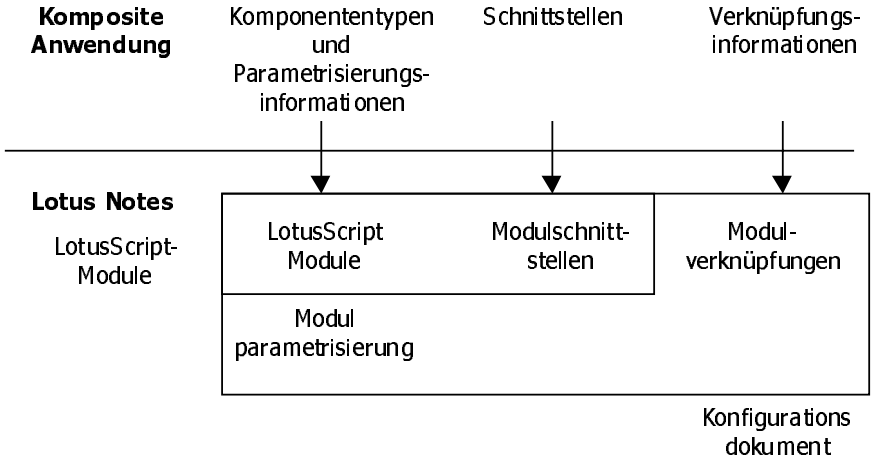


Abbildung 153: Komposite Anwendungen in Lotus Notes

Im ersten Schritt wurden Nachbildungen des Konzepts der Komponente und des komponentenorientierten Rahmenwerks mit den Mitteln von Lotus Notes geschaffen. Diese besitzen zwar nicht alle Eigenschaften von Komponenten und komponentenorientierten Rahmenwerken, für die Ziele des Projektes reichten sie jedoch aus. Module der in Lotus Notes vorhandenen Skriptsprache LotusScript [Note] wurden an Stelle von Komponenten verwendet, gleiches gilt für die Schnittstellen der Module. Die Registratur komponentenorientierter Rahmenwerke wurde mit Hilfe der in Lotus Notes vorhandenen Verzeichnisdienste nachgebildet. Auf dieser Basis konnte dann eine Umsetzung des Konzepts der kompositen Anwendung durchgeführt werden. Die Verknüpfungen in einer kompositen Anwendung und die Konfiguration von Diensten wurde durch geeignete Verknüpfungen und Parametrisierungen der LotusScript-Module bewerkstelligt. Die dazu notwendigen Informationen wurden in einem sogenannten Konfigurationsdokument festgehalten, das die Rolle eines Anwendungsdokumentes einnimmt.

Auf der Basis dieser Vorarbeiten konnte dann die Umsetzung der aspektelementorientierten Schemarepräsentation durchgeführt werden, was in Abbildung 154 veranschaulicht ist. Die Repräsentationselemente der aspektelementorientierten Schemarepräsentation werden durch LotusScript-Module dargestellt. Ihre Anpassung geschieht durch die Parametrisierung der Module. Die Verbindungspunkte der aspektelementorientierten Schemarepräsentation wurden auf dementsprechende Schnittstellen der Module umgesetzt. Die Verbinder wurden auf Verknüpfungen der Module abgebildet, die zusammen mit der Modulparametrisierung im Konfigurationsdokument gespeichert werden.

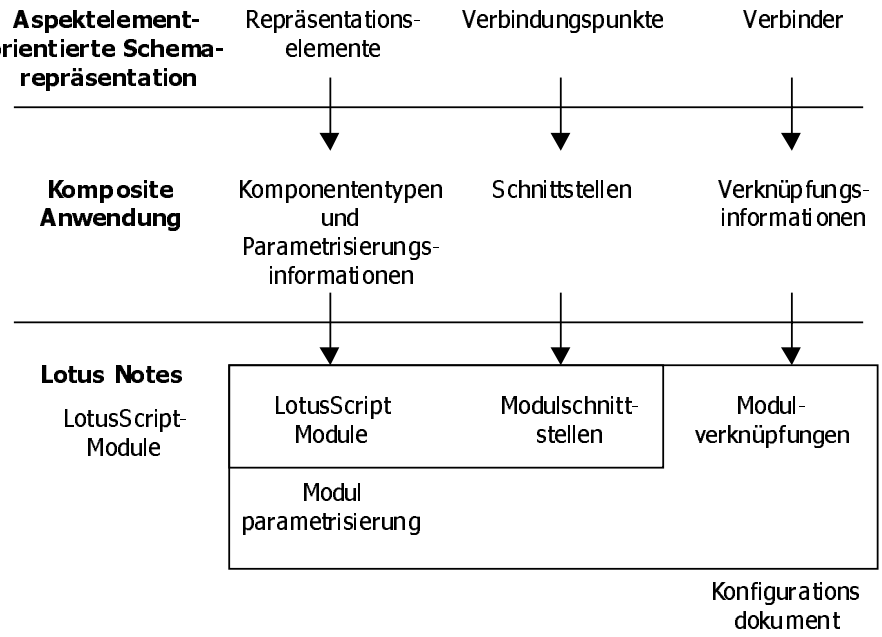


Abbildung 154: Umsetzung der aspektelementorientierten Schemarepräsentation

12.1.3 Bewertung

Die im Rahmen des Projektes entwickelte informationstechnische Unterstützung des Geschäftsprozesses CSPA erfüllte voll die Anforderungen und wird vom Auftraggeber operativ eingesetzt. Durch die Verwendung einer aspektelementorientierten Schemarepräsentation konnte eine flexible Umsetzung von Workflow-Schemaänderungen erreicht werden und es wurden ausschließlich von Lotus Notes bereitgestellte Mittel verwendet.

Darüber hinaus konnten folgende Erfahrungen gesammelt werden. Der Frage des Aufwands für die Abbildung von Geschäftsprozeß- auf Workflow-Schemata wurde zu Projektanfang zu wenig Aufmerksamkeit geschenkt. Aus diesem Grunde wurde eine Gesamtabbildung der Geschäftsprozeßschemata als geeignet angesehen. Im Rahmen der praktischen Evaluierung tauchte allerdings das Problem auf, daß diese Gesamtabbildung sehr aufwendig ist. Hieraus ergab sich die Anforderung der inkrementellen Abbildung von Geschäftsprozeß- auf Workflow-Schemata. Diese Anforderung ging in die Aufgabenstellung des CompFlow Projekts ein und führte zur Schaffung eines inkrementellen Abbildungsverfahrens, das im SOMFlow-Projekt weiter verfeinert wurde. Ebenfalls stellte sich die Notwendigkeit heraus, die Menge der unterstützen Workflow-Schemaelemente zu erweitern. D.h. das Workflow-Modell darf nicht statisch repräsentiert werden, sondern muß die dynamische Erweiterung um zusätzliche Modellelemente ermöglichen.

12.2 CompFlow

12.2.1 Projektziel

Das CompFlow-Projekt hatte das Ziel, die informationstechnische Unterstützung weitreichender Prozesse auf der Basis der Enterprise JavaBeans Technologie [EJB] bereitzustellen [Würg98], [Davi99] und ein inkrementelles Abbildungsverfahren für Geschäftsprozeßschemata in der SOM-Notation zu schaffen [Heri98]. In ihm kommen alle entwickelten Konzepte zum Einsatz. Die inkrementelle Abbildung wurde im SOMFlow-Projekt weiterentwickelt und wird daher erst dort vorgestellt.

Die Darstellung des CompFlow-Projekts konzentriert sich deswegen auf die Implementierung der aspektelementorientierten Schemarepräsentation und Umsetzung des Workflow-Modell-Wörterbuchs. Ausgangspunkt ist die Vorstellung der Enterprise JavaBeans Technologie. Dies geschieht, um eine Grundlage für die Darstellung der implementierten Konzepte zu legen.

12.2.2 Die Enterprise JavaBeans Technologie

Die Enterprise JavaBeans Technologie wurde ausgewählt, da sie als Java-basierte Technologie eine hohe Plattformunabhängigkeit aufweist und sich daher besonders gut für die bei weitreichenden Geschäftsprozessen anzutreffenden heterogenen Umgebungen eignet. Sie wird unter der Maßgabe vorgestellt, wie mit ihr die in Kapitel 9 beschriebenen Konzepte der Komponente und des komponentenorientierten Rahmenwerks verwirklicht werden. Da in der Enterprise JavaBeans Technologie (EJB) keine Implementierung kompositer Anwendungen vorhanden ist, wird diese später ergänzt. Als Ergebnis steht ein vollständiges Komponentensystem zur Verfügung.

Die Bestandteile der Enterprise JavaBeans Technologie implementieren mit einer Reihe von unterstützenden Technologien das Konzept der Komponente und des komponentenorientierten Rahmenwerks. Um Mißverständnisse zu vermeiden, welche der Bestandteile gemeint sind, gilt folgende Regelung. Von Enterprise JavaBeans, kurz EJB, wird die Rede sein, wenn Bestandteile der Enterprise JavaBeans Technologie gemeint sind, die das Konzept des komponentenorientierten Rahmenwerks unterstützen. Von Enterprise JavaBeans Komponenten, kurz EJB-Komponenten, wird die Rede sein, wenn es sich um Bestandteile der Enterprise JavaBeans Technologie handelt, die das Konzept der Komponente unterstützen.

Bei den unterstützenden Technologien handelt es sich um Technologien, die eigenständig existieren, aber im Zusammenspiel mit Enterprise JavaBeans Teilfunktionalitäten eines komponentenorientierten Rahmenwerks übernehmen. Der Java Naming and Directory Service [JNDI] ist eine solche unterstützende Technologie, die eigenständig ist, aber im Zusammenspiel mit EJB die Rolle der Registratur eines komponentenorientierten Rahmenwerks übernimmt.

12.2.2.1 EJB als komponentenorientiertes Rahmenwerk

In diesem Abschnitt erfolgt die Zuordnung von EJB und unterstützenden Technologien zu den Teilkonzepten komponentenorientierter Rahmenwerke. Die Ergebnisse sind in Abbildung 155 dargestellt.

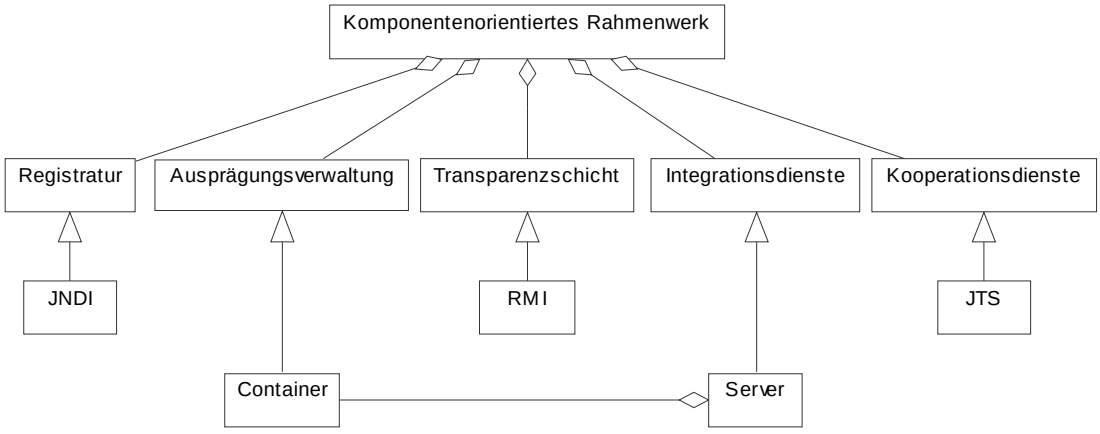


Abbildung 155: EJB als komponentenorientiertes Rahmenwerk

In Kapitel 9 wurde als zentrale Aufgabe komponentenorientierter Rahmenwerke das Lebenszyklusmanagement von Komponenten identifiziert. Dazu wird eine Ausprägungsverwaltung eingesetzt, deren Aufgabe es ist, Komponentenausprägungen transparent für die anfordernde Anwendung zu erzeugen und wieder zu zerstören. Unterstützt wird sie dabei durch eine Transparenzschicht, die einen transparenten Zugriff auf die Komponentenausprägungen ermöglicht. Die zur Bildung von Ausprägungen zur Verfügung stehenden Komponenten werden in einer Registratur verwaltet. Kooperationsdienste regeln die Zusammenarbeit von Komponentenausprägungen beispielsweise im Rahmen von Transaktionen. Integrationsdienste übernehmen die Integration von Diensten, die nicht von Komponenten bereitgestellt werden.

Die Rolle der Ausprägungsverwaltung eines komponentenorientierten Rahmenwerks übernehmen die sogenannten Container in EJB. Nach außen sichtbar ist der Container jedoch nicht. Die Anforderung von Komponentenausprägungen geschieht über die sogenannte EJBHome-Schnittstelle, die im Abschnitt 12.2.2.2 beschrieben wird. Über sie können EJB-Ausprägungen erzeugt, gelöscht oder existierende Ausprägungen an Hand eines Schlüssels gesucht werden. In einem EJB-Container können sich verschiedene Typen von EJB-Komponenten befinden. Die in einem EJB-Container befindlichen Komponentenausprägungen sind für den Klienten nicht direkt zugänglich, sondern nur über einen Stellvertreter, der vom Container zur Verfügung gestellt wird.

Die Rolle der Registratur nimmt der Java Naming- und Directory Service (JNDI) [JNDI] ein. Sie und mit ihr in Verbindung stehende Konzepte sind in Abbildung 156 dargestellt.

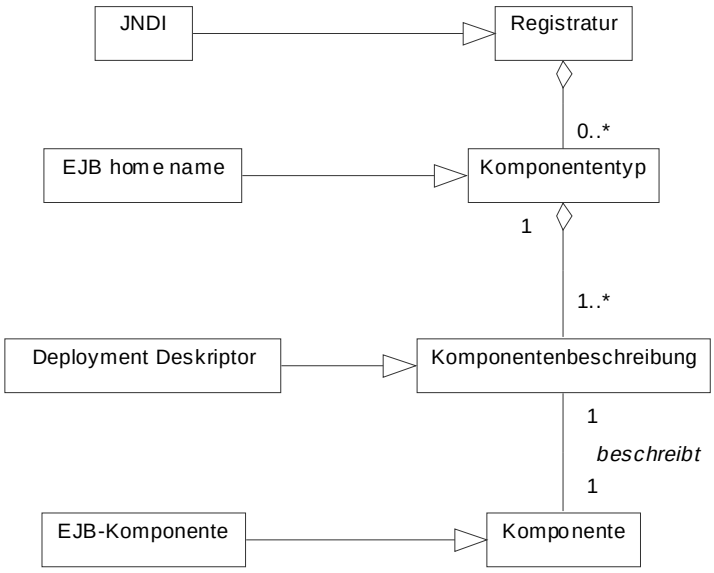


Abbildung 156: EJB-Registratur

Der sogenannte „EJB-home name“ stellt die Bezeichnung des Komponententyps in JNDI dar. Ihm ist genau ein sogenannter Deployment-Deskriptor zugeordnet, der eine Art erweiterter Komponentenbeschreibung darstellt, die außerdem auch Spezialisierungsinformationen enthält. Dies bedeutet jedoch, daß es für jeden Komponententyp nur eine Komponente und besonders nur einen Satz von Spezialisierungsinformationen geben kann. Mit Hilfe des Deployment Deskriptors wird beispielsweise das transaktionale Verhalten der EJB-Komponente festgelegt. Ebenfalls kann bestimmt werden, wer die EJB-Komponente nutzen darf. Jeder Deployment-Deskriptor ist genau einer EJB-Komponente zugeordnet. Gegen über dem in Abbildung 80 dargestellten allgemeinen Modell einer Registratur fehlen in EJB lediglich die Schnittstellenbeschreibungen.

Der Remote Method Invocation Mechanismus (RMI) [RMI] übernimmt die Aufgabe der Transparenzschicht. Durch ihn kann auf EJB-Komponenten transparent vom Ort des Containers, in der sie sich befindet, zugegriffen werden.

Ein oder mehrere Container werden in einem Server zusammengefaßt. Dieser stellt die Verbindung zu Datenbanken her. Seine Rolle ist daher am ehesten mit der der Integrationsdienste zu vergleichen.

Schließlich existieren auch Kooperationsdienste wie beispielsweise der Java Transaction Service (JTS) [JTS]. Er ermöglicht es, eine oder mehrere EJB-Komponentenausprägungen in einer Transaktion ablaufen zu lassen, wobei drei Ausformungen möglich sind: Durch den Dienstnehmer verwaltete Transaktionen („Client managed“) werden durch den Dienstnehmer explizit begonnen und beendet. Durch den Container verwaltete Transaktionen („Container managed“) werden transparent für die EJB-Komponentenausprägungen verwaltet. Die dritte Art von unterstützten Transaktionen sind direkt durch die EJB-Komponentenausprägungen verwaltete Transaktionen, bei denen diese selbst Beginn und Ende der Transaktion steuern.

12.2.2.2 EJB-Komponenten

Die EJB-Komponenten entsprechen in weiten Zügen dem in Kapitel 9 vorgestellten Referenzmodell. Sie besitzen mehrere Schnittstellen sowie Spezialisierungs- und Introspektionsmechanismen. Die Zuordnung der einzelnen EJB-Komponentenbestandteile zum in Kapitel 9 vorgestellten Referenzmodell ist in Abbildung 157 dargestellt.

Die Spezialisierung von EJB-Komponenten erfolgt auf der Basis von Spezialisierungsinformationen, die im Deployment-Deskriptor abgelegt sind. Da für jeden Komponententyp nur ein Deployment-Deskriptor angebbar ist, kann nur eine generelle Spezialisierung durchgeführt werden, gemäß der Klassifikation in Kapitel 9. Alle Ausprägungen erhalten die gleichen Spezialisierungsinformationen. Mit den im Deployment-Deskriptor enthaltenen Spezialisierungsinformationen können Parameter festgelegt werden, aber auch Referenzen auf andere Enterprise JavaBeans bestimmt werden. Letztere werden als EJB-Referenzen bezeichnet.

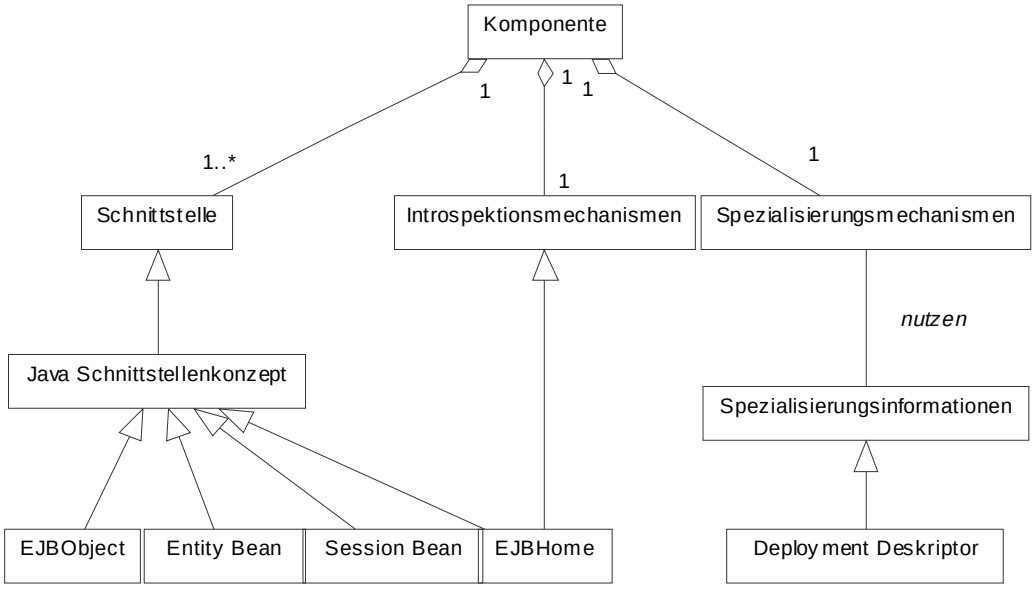


Abbildung 157: EJB-Komponenten

EJB-Komponenten können nicht nur, sondern sie müssen sogar mehrere Schnittstellen besitzen. So müssen sie mindestens die Schnittstellen EJBHome, EJBObject sowie die SessionBean- oder die Entity-Bean-Schnittstelle implementieren. EJBHome ist für die Erzeugung von EJB-Ausprägungen, das Auffinden von Ausprägungen und das Löschen von EJB-Ausprägungen zuständig. Ebenfalls wird die Introspektion einer EJB-Komponente über die EJBHome-Schnittstelle abgewickelt. Die Schnittstelle EJBObject dient zum Zugriff auf die eigentliche Funktionalität der EJB-Komponente. In Enterprise JavaBeans werden zwei Arten von Komponenten unterschieden, Entity-Beans und Session-Beans. Entity-Beans unterstützen die Entity-Bean Schnittstelle und Session-Beans die SessionBean Schnittstelle.

Entity-Beans sind persistent und repräsentieren Entitäten, die durch einen Primärschlüssel identifiziert werden. Sie besitzen einen von zwei Persistenzmechanismen. Die Bean-managed-persistence bedeutet, daß die Entity-Bean selbst für die Verwaltung der Persistenz verantwortlich ist. Die Container-Managed-Persistence wird durch den Container, in dem sich die Entity-Bean befindet, bereitgestellt. Dieser verknüpft die Attribute der Entity-Bean transparent mit einer Datenbank oder anderen Datenquellen. Bei der Bean-managed-persistence muß die Entity-Bean zwei Methoden bereitstellen, mit denen der Container das Speichern und Laden ihres Zustandes anfordern kann.

Die Lebensdauer von Entity-Beans kann die Lebensdauer der ihnen zu Grunde liegenden Daten nicht übertreffen. Entity-Beans können von mehreren Dienstnehmern gleichzeitig benutzt werden. Eine Ausprägung einer Entity-Bean kann sich in zwei Zuständen befinden: Im Instanzenpool und bereit. Der Zustand im Instanzenpool bedeutet, daß die Ausprägung existiert, aber noch nicht mit bestimmten Daten verbunden ist. Bereit bedeutet, daß die Ausprägung mit den ihr zugehörigen Daten verbunden ist. Der Übergang vom Zustand „im Instanzenpool“ in den Zustand „bereit“ geschieht, wenn ein Dienstnehmer die Ausprägung anfordert.

Im Gegensatz zu Entity-Beans sind Session-Beans kurzlebig und immer nur einem Dienstnehmer zugeordnet. Sie sind nicht persistent und können auch nicht wiederhergestellt werden. Eine Sonderform sind die Stateless Session-Beans. Sie besitzen keinen Zustand und sind daher leichter für den Container zu verwalten. Dieselbe Ausprägung einer Session-Bean kann von verschiedenen Klienten genutzt werden.

12.2.3 Lösungsansatz

12.2.3.1 Komposite Anwendungen auf Enterprise JavaBeans

Durch die Enterprise JavaBeans Technologie und unterstützende Technologien werden Implementierungen des Konzepts der Komponente und des komponentenorientierten Rahmenwerks bereitgestellt. Nicht unterstützt wird jedoch das Konzept der kompositen Anwendung. Zur Implementierung des Konzepts der kompositen Anwendung mit Hilfe der Enterprise JavaBeans Technologie wird zunächst der Spezialisierungsmechanismus von EJB so erweitert, daß eine registraturbasierte Spezialisierung möglich ist. Auf diese Weise kann eine EJB-Komponente in mehreren kompositen Anwendungen verwendet werden.

Eine registraturbasierte Spezialisierung ist in EJB realisierbar, wenn man die Kontextinformation durch die Bildung von Untertypen erfaßt. Anstatt die Deployment-Deskriptoren direkt bei einem Komponententyp zu speichern, wird für jeden Kontext ein Komponentenuntertyp gebildet. Auf diese Weise können unterschiedliche Spezialisierungsinformationen in Form von Deployment Deskriptoren für unterschiedliche Kontexte angegeben werden. Veranschaulicht ist dies für den Komponententyp k1 in Abbildung 158. Von ihm werden Untertypen für drei Kontexte 1, 2 und 3 gebildet, indem Untertypen der Komponente k1 gebildet werden, die als k11, k12 und k13 gekennzeichnet werden.

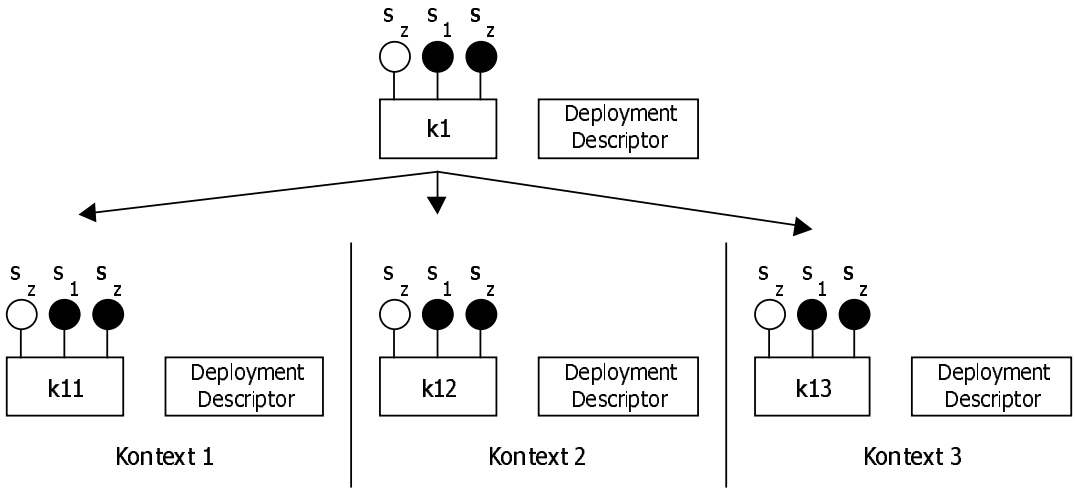


Abbildung 158: Registraturbasierte Spezialisierung in EJB

Wie in Abschnitt 12.2.2.2 dargestellt, ist der Deployment-Deskriptor für die Speicherung der Spezialisierungsinformationen einer EJB-Komponente zuständig. Daher wird für ihn ein Konzept zur Speicherung der Parametrisierungs- und Spezialisierungsinformationen geschaffen.

Die Speicherung von Parametrisierungsinformationen durch den Deployment-Deskriptor ist direkt möglich, solange die vom Deployment-Deskriptor angebotenen Datentypen String, Integer, Boolean, Double und Float ausreichen. Ein Beispiel hierfür ist in Abbildung 159 dargestellt. Es zeigt einen Ausschnitt des in XML [XML] dargestellten Deployment-Deskriptors für die Festlegung des Werts 5000 für den Parameter „Auftragswert“. In (1) wird der Parametername angegeben. In (2) wird der Parameterwert bestimmt.

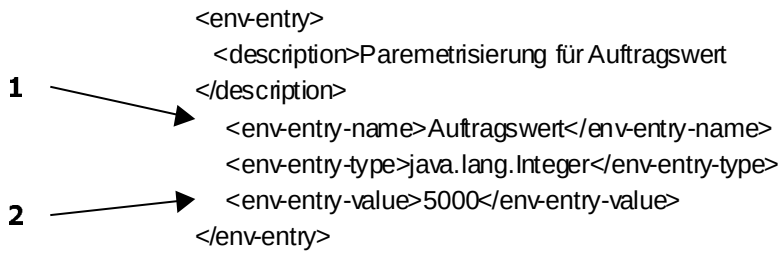


Abbildung 159: Verwendung des Deployment-Deskriptors für Parametrisierungsinformationen

Durch den Deployment-Deskriptor können auch die Verknüpfungsinformationen einer kompositen Anwendung erfaßt werden. In Abbildung 160 ist dargestellt, wie der Ausschnitt eines Deployment-Deskriptor beschaffen ist, der zur Wiedergabe einer Verknüpfung mit einer EJB-Komponente des Typs k1 über die Schnittstelle s_1 dient. In (1) ist der Typ der Zielkomponente wiedergegeben. Die Zielschnittstelle ist in (2) festgelegt.

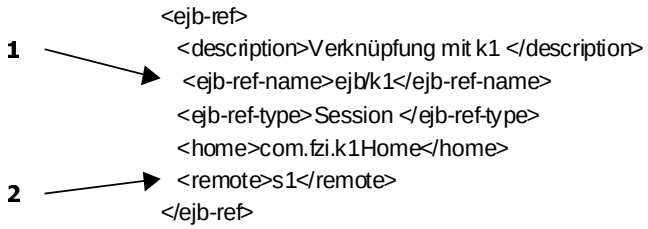


Abbildung 160: Verwendung des Deployment-Deskriptors für Verknüpfungsinformationen

12.2.3.2 Implementierung der aspektelementorientierten Schemarepräsentation

Aufgabe bei der Implementierung der aspektelementorientierten Schemarepräsentation ist es, die Bestandteile der aspektelementorientierten Schemarepräsentation, also Repräsentationselemente, Verbinder und Verbindungspunkte, auf Bestandteile einer kompositen Anwendung abzubilden, die mit Hilfe von EJB realisiert ist. In Abschnitt 10.1.1 wurde eine Abbildung von Bestandteilen der aspektelementorientierten Schemarepräsentation auf Bestandteile kompositer Anwendungen durchgeführt, eine Zusammenfassung findet sich in Abbildung 161.

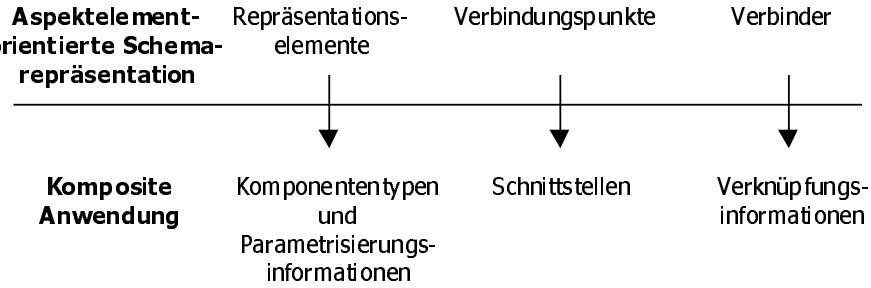


Abbildung 161: Die aspektelementorientierte Schemarepräsentation als komposite Anwendung

In Abschnitt 12.2.3.1 wurde gezeigt, wie mit den Mitteln der EJB-Technologie komposite Anwendungen gebildet werden können. Daher können nun beide Lösungen zusammengeführt werden und die Implementierung der aspektelementorientierten Schemarepräsentation auf der Basis einer kompositen Anwendung durchgeführt werden, die mit Hilfe der EJB-Technologie bereitgestellt wird.

Bei der Abbildung der aspektelementorientierten Schemarepräsentation werden Repräsentationselemente und Verbindungspunkte auf Komponententypen und Schnittstellen einer kompositen Anwendung abgebildet. Wie in Abschnitt 8.2.2 dargestellt, gibt es zwei grundsätzliche Arten von Verbindungspunkten: Temporale und nicht-temporale Verbindungspunkte. Temporale Verbindungspunkte dienen als Anknüpfungspunkt für temporale Verbinder, nicht-temporale Verbindungspunkte dienen als Anknüpfungspunkte für nicht-temporale Verbinder. Umgesetzt werden diese Verbindungspunkte in der kompositen Anwendung durch die Schnittstelle s_z für temporale Verbindungspunkte und individuelle Schnittstellen für nicht-temporale Verbindungspunkte. In Abhängigkeit ob es sich um einen ein- oder ausgehenden Verbindungspunkt handelt, werden diese durch eine ein- oder ausgehende Schnittstelle umgesetzt.

Zuerst wird geklärt, wie eine Abbildung auf Entity-Beans und Session-Beans erfolgen kann. Hierzu wird die in Kapitel 10 durchgeführte Einteilung der Komponenten auf der Basis ihrer ein- und ausgehenden Schnittstellen herangezogen. Die für die Implementierung relevanten Komponentenarten sind in Tabelle 12 dargestellt. In ihr ist das Vorhandensein einer Schnittstelle mit einem Pluszeichen dargestellt. Das Fehlen einer Schnittstelle ist mit einem Minuszeichen wiedergegeben. Ist das Vorhandensein der Schnittstelle freigestellt, so ist ein Kreiszeichen eingetragen.

	s_z eingehend	s_z ausgehend	individuelle Schnittstelle eingehend	individuelle Schnittstelle ausgehend
Initiale Komponenten	-	+	-	-
Finale Komponenten	+	-	-	-
Temporale Komponenten	+	+	-	o
Dienstkomponenten	-	-	+	+
	-	-	+	-

Tabelle 12: Komponentenarten zur Abbildung auf EJB

Die initialen und finalen Komponenten sind genauso wie die temporalen Komponenten zeitlich nur begrenzt aktiv. Sie können zwar einen Zustand besitzen, dieser braucht aber nicht persistent zu sein, da die Speicherung persistenter Informationen Angelegenheit des Datenaspekts ist. Eine Speicherung von Informationen in einem temporal orientierten Repräsentationselement, das nicht dem Datenaspekt zugehörig ist, würde die Aspektseparation verletzen. Aus diesem Grund empfiehlt sich die Verwendung von Session-Beans für die Implementierung von initialen, finalen und temporal orientierten Komponenten.

Bei den Dienstkomponenten kann die Entscheidung zwischen Session- oder Entity-Beans nur in Abhängigkeit vom Aspektelement getroffen werden, das von der Komponente unterstützt wird. Dies muß von Fall zu Fall entschieden werden, so daß keine allgemeinen Regeln angebbar sind. Eine Übersicht der Implementierung der verschiedenen Komponentenarten auf Entity- oder Session-Beans gibt Abbildung 162.

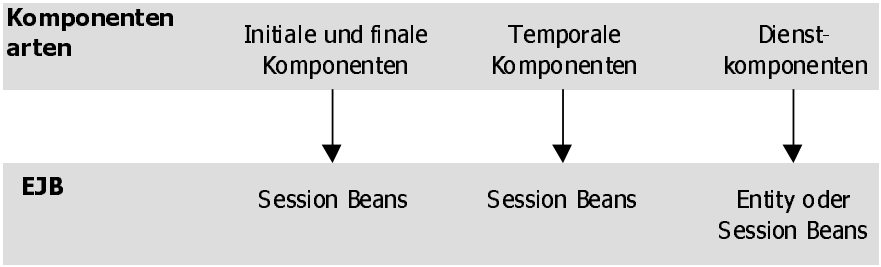


Abbildung 162: Implementierung von Komponententypen durch EJB-Komponenten

Die Verbinder der dynamischen Schemarepräsentation dienen zur Verbindung von Repräsentationselementen. Es lassen sich zwei Arten von Verbindern unterscheiden. Temporale Verbinder geben zeitliche Abhängigkeiten zwischen den Repräsentationselementen wieder. Nicht-temporale Verbinder repräsentieren Dienstnehmer-/Dienstgeberbeziehungen.

Verbinder werden gemäß Abbildung 161 auf Verknüpfungsinformationen einer kompositen Anwendung abgebildet. Der hierzu in Abschnitt 12.2.3.1 identifizierte Mechanismus, die Angabe von EJB-Komponentenreferenzen im Deployment-Deskriptor, kann temporale- und nicht-temporale Verbinder gleichermaßen abbilden. In ihm ist es möglich, die Zielschnittstelle anzugeben, so daß für die Wiedergabe temporaler Verbinder die Schnittstelle s_z und für nicht-temporale Verbinder die individuelle Schnittstelle angegeben werden kann.

Zusammenfassend werden die Bestandteile der aspektelementorientierten Schemarepräsentation wie in Abbildung 163 auf EJB abgebildet. Die Repräsentationselemente der aspektelementorientierten Schemarepräsentation werden entsprechend der Klassifikation in Tabelle 12 auf Entity- bzw. Session-Beans und Deployment-Deskriptoren abgebildet. Die Verbindungspunkte der aspektelementorientierten Schemarepräsentation werden auf die Schnittstelle s_z und individuelle Schnittstellen von EJB-Komponenten abgebildet. Die Verbinder der aspektelementorientierten Schemarepräsentation werden auf Deployment-Deskriptoren abgebildet.

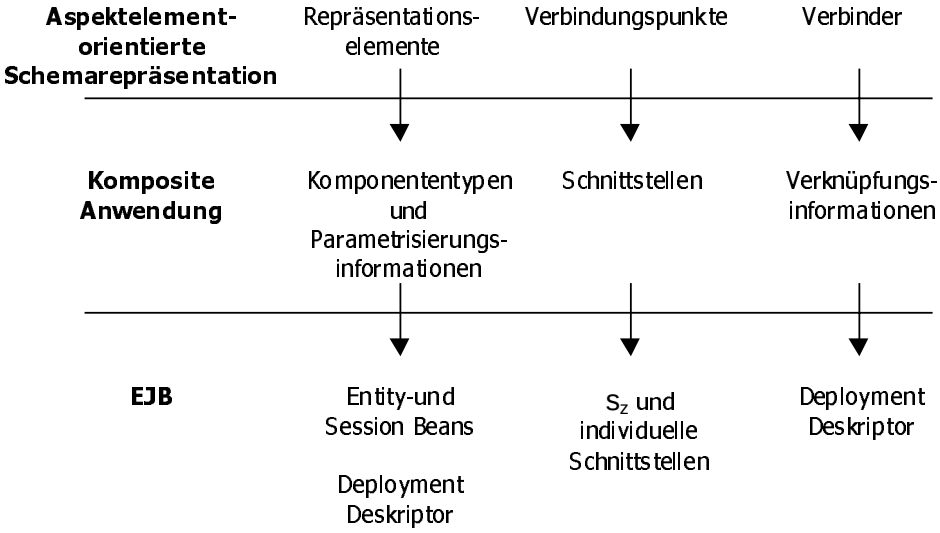


Abbildung 163: Implementierung der aspektelementorientierten Schemarepräsentation durch EJB

12.2.3.3 Implementierung des Beispiel-Workflows

Im Rahmen des CompFlow-Projekts wurde auch die Implementierung des bekannten Beispielschemas der Auftragserfassung und -prüfung durchgeführt. Dazu wird das Vorgehen aus Abschnitt 10.1.3 von Seite 131 aufgegriffen. Dies bedeutet, daß zunächst unter Anwendung der entwickelten Abbildungsregeln EJB-Komponententypen spezifiziert werden. Im zweiten Schritt

werden Komponentenuntertypen entwickelt, um eine registraturbasierte Spezialisierung zu ermöglichen. Den dritten Schritt bildet die Schaffung von Deployment-Deskriptoren.

Den ersten Schritt bildet die Darstellung der zur Umsetzung des Beispielschemas zu verwendenden EJB-Komponenten. Sie sind in Abbildung 164 unter Angabe ihrer Aspektzugehörigkeit dargestellt.

Für die Umsetzung von Schemaelementen des Beispiels, die dem Operationsaspekt zugehörig sind, werden die Komponententypen oa, od, os und ow benötigt. Sie enthalten die Schnittstelle s_z und die Schnittstelle s_1. Letztere ist eine ausgehende Schnittstelle, die zum Ansprechen der benötigten Daten in dem Komponententyp da, also dem Auftragsspeicher, dient. Für die Bestandteile des Schemas, die dem Kontrollaspekt zugehörig sind, werden die Komponenten des Typs ks und kj verwendet. Der Komponententyp ks ist ein Gabelungselement. Er besitzt eine eingehende Schnittstelle s_z zu seiner Aktivierung und zwei ausgehende Schnittstellen. Die erste, s_z, dient zur Aktivierung der Komponente. Mit der zweiten, s_1, wird das als Entscheidungskriterium dienende Datum erfragt. Mit Hilfe von Parametrisierungsinformationen wird Name und Wert des als Entscheidungskriteriums dienenden Wertes angegeben. Die in den Verknüpfungsinformationen von s_z angegebenen Schnittstellen werden in Abhängigkeit vom Ergebnis des Tests aktiviert. Der Komponententyp kj ist ein Verbindungselement. Er besitzt eine eingehende Schnittstelle s_z und eine ausgehende Schnittstelle s_z. Zur Speicherung der Auftragsdaten wird der Komponententyp da verwendet. Er verfügt nur über eine eingehende Schnittstelle s_1. Der Anschaulichkeit halber wird angenommen, daß durch sie direkt der Zugriff auf Daten erfolgen kann, ohne daß weitere Komponenten verwendet werden müssen.

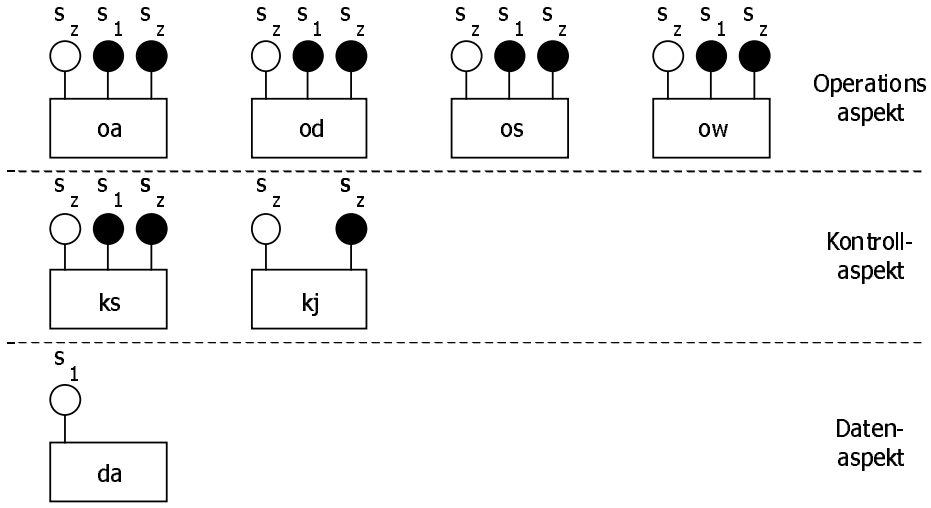


Abbildung 164: EJB-Komponenten zur Implementierung des Beispielschemas

An Hand dieser Informationen kann bestimmt werden, welche der Komponenten durch welche Art von EJB-Komponenten, also Entity- oder Session-Bean, implementiert werden sollen. Dabei kommen die in Abschnitt 10.1.2 entwickelten Regeln zum Einsatz. Die Komponenten oa, od, os und ow sind der Gruppe der temporalen Komponenten zugehörig. Daher sind diese Komponenten als Session-Beans zu implementieren. Die Komponenten ks und kj sind ebenfalls der Gruppe der temporalen Komponenten zugehörig und müssen daher als Session-Beans implementiert werden. Die Komponente da besitzt nur eine eingehende individuelle Schnittstelle und ist daher der Gruppe der Dienstkomponenten zugehörig. Deswegen ist ihre Ausführung als Entity- oder Session Bean nur auf Grund ihrer Semantik entscheidbar. Da sie aber ein Element des Datenaspekts unterstützt, muß sie als Entity-Bean implementiert werden.

Da EJBs standardmäßig nur eine generelle Spezialisierung kennen, wurde in Abschnitt 12.2.3.1 ein Konzept entwickelt, um durch die Bildung von Komponentenuntertypen eine registraturbasierte Spezialisierung zu ermöglichen. Daher müssen vor der Schaffung von Deployment-Deskriptoren Komponentenuntertypen gebildet werden. Diese Komponentenuntertypen tragen die Bezeichnung oa1, os1, od1, ow1, ks1, kj1 und da1. Die im folgenden gebildeten Spezialisierungsinformationen in Form von Deployment-Deskriptoren werden diesen Untertypen zugeordnet. Spezialisierungsinformationen anderer Kontexte werden dann in Deployment-Deskriptoren abgelegt, die beispielsweise den Komponentenuntertypen oa2, os2, usw. zugeordnet werden.

Der nächste Schritt der Implementierung ist die Erzeugung von Deployment-Deskriptoren für die gebildeten Komponentenuntertypen. Dazu müssen die in Tabelle 9 enthaltenen Spezialisierungsinformationen auf dementsprechende Deployment-Deskriptoren umgesetzt werden. Die Ergebnisse sind in Tabelle 13 wiedergegeben.

Komponententyp	Spezialisierungsinformationen
oa1	$s_{z,}$ ks1; $s_{1,}$ da1
ks1	$s_{z,}$ os1; $s_{z,}$ od1; $s_{1,}$ da1; Auftragswert, 5000
os1	$s_{z,}$ kj1; $s_{1,}$ da1
od1	$s_{z,}$ kj1; $s_{1,}$ da1
kj1	$s_{z,}$ ow
ow1	$s_{z,}$ e

Tabelle 13: Umzusetzende Spezialisierungsinformationen

Exemplarisch wird an den Spezialisierungsinformationen für den Komponententyp ks1 die Erzeugung der Deployment-Deskriptoren durchgeführt. Die Spezialisierungsinformationen enthalten eine Verknüpfung mit dem Komponententyp os1 über die Schnittstelle s_z, dem Komponententyp od1 über die Schnittstelle s_z, und dem Komponententyp da1 über die Schnittstelle s_1. Außerdem ist eine Parametrisierungsinformation für den Parameter Auftragswert mit dem Wert 5000 vorhanden.

Zunächst werden die Eintragungen im Deployment-Deskriptor für die Verknüpfungsinformationen betrachtet. Die Abbildung 165 enthält die Eintragungen für die Verknüpfungen mit den Komponententypen os1, od1 und da1. In den mit (1) gekennzeichneten Zeilen befindet sich jeweils die Angabe des Zielkomponententyps. Die Zielschnittstelle wird in den mit (2) gekennzeichneten Zeilen angegeben.

```
        <ejb-ref>
            <description>Komponente os </description>
(1)     <ejb-ref-name>ejb/os1</ejb-ref-name>
            <ejb-ref-type>Session </ejb-ref-type>
            <home>com.fzi.ks1Home</home>
(2)         <remote>sz</remote>
        </ejb-ref>

        <ejb-ref>
            <description>Komponente os </description>
(1)         <ejb-ref-name>ejb/od1</ejb-ref-name>
            <ejb-ref-type>Session </ejb-ref-type>
            <home>com.fzi.ks1Home</home>
(2)         <remote>sz</remote>
        </ejb-ref>

        <ejb-ref>
            <description>Komponente os </description>
(1)     <ejb-ref-name>ejb/da1</ejb-ref-name>
            <ejb-ref-type>Session </ejb-ref-type>
            <home>com.fzi.ks1Home</home>
(2)     <remote>s1</remote>
        </ejb-ref>
```

Abbildung 165: Verknüpfungsinformationen

Im nächsten Schritt wird der Eintrag für die Darstellung der Parametrisierungsinformationen erstellt. In Abbildung 166 ist dargestellt, wie dem Parameter Auftragswert der Wert 5000 zugewiesen wird. In (1) wird der Parametername angegeben, in (2) der Parameterwert bestimmt.

```
        <env-entry>
            <description>Auftragswert</description>
(1)         <env-entry-name>Auftragswert</env-entry-name>
            <env-entry-type>java.lang.Integer</env-entry-type>
(2)         <env-entry-value>5000</env-entry-value>
        </env-entry>
```

Abbildung 166: Parametrisierungsinformationen

12.2.3.4 Implementierung des Workflow-Modell-Wörterbuchs

Die Implementierung des Workflow-Modell-Wörterbuchs konzentrierte sich auf die eigentliche Herausforderung, nämlich die Integration der im Java Naming und Directory Service vorhandenen Informationen. In Abschnitt 10.2.2 wurde ein Integrationsmodell für Komponentensysteme vorgestellt. Es galt daher das Integrationsmodell für die Verwendung mit EJB anzupassen.

Um diese Anpassung für EJB durchzuführen, mußten zunächst die Informationen im JNDI analysiert werden. Dabei zeigte sich, daß EJB praktisch alle Informationen anbietet, die im Integrationsmodell verwendet werden. In Abbildung 167 ist das Integrationsmodell dargestellt. So können sich Ressourcenbeschreibungen auf Informationen aus dem Deployment Deskriptor stützen und beispielsweise den Komponententyp auswerten. Durch Auswertung der Schnittstellenbeschreibungen läßt sich eine Grundlage für die Dienstbeschreibung gewinnen.

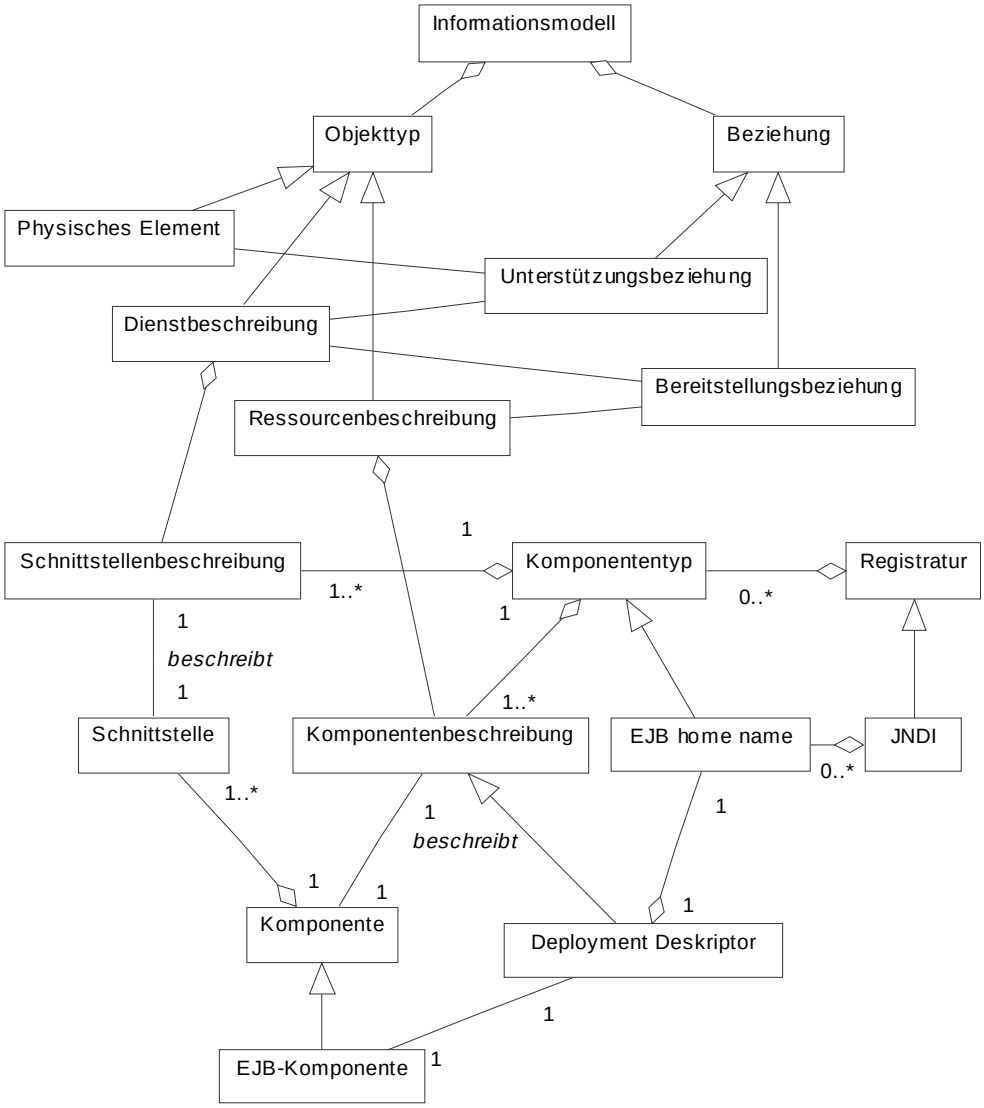


Abbildung 167: Integrationsmodell für die EJB-Registratur

Der Einsatz des Workflow-Modell-Wörterbuchs wurde im CompFlow-Projekt am Beispiel des Hinzufügens einer Transaktionsunterstützung praktisch evaluiert. Die Transaktionsunterstützung in Workflows ist notwendig, um Teile oder komplette Workflows unter Einhaltung von Korrektheitskriterien ablaufen zu lassen [GeHo94], [RuSh94], [KaRa96]. Die restriktivste Form ist die der klassischen ACID-Transaktion [LaLo95]. Für Workflows sind jedoch häufig andere Formen erforderlich, da sich Workflows über lange Zeiträume erstrecken können, was unter Beibehaltung der klassischen ACID-Transaktionen zu Problemen führt. Aus diesem Grund entstand die Notwendigkeit, nicht nur ACID-Transaktionen, sondern sogenannte erweiterte Transaktionsmodelle [AAAK96], [Elma92] zu unterstützen. Erweiterte Transaktionsmodelle erlauben beispielsweise mehrere Transaktionen zu gruppieren, oder aber Abschwächungen des ACID-Paradigmas, beispielsweise durch Reduzierung der Isolation der Transaktionen, durchzuführen.

Für die Erfassung der Transaktionsunterstützung im Workflow-Modell wurde dem Zwei-Ebenen-Workflow-Metamodell entsprechend ein logischer und ein physischer Aspekt eingeführt: Der sogenannte Korrektheitsaspekt auf der logischen Ebene und der Transaktionsaspekt auf der physischen Ebene. Durch den Korrektheitsaspekt wird angegeben, ob und unter welcher Art von Transaktionsunterstützung der Workflow oder Teile von ihm ausgeführt werden. Im Transaktionsaspekt werden die dafür notwendigen informationstechnischen Ressourcen beschrieben.

Bei der Gestaltung des Korrektheits- und des Transaktionsaspekts stellt sich die Frage, wodurch sich die Aspektelemente definieren. Hierfür wurden zwei Alternativen entwickelt. Die erste Lösung ist, für jede Transaktionsform ein eigenes Aspektelement zu verwenden. Dies wäre jedoch zu inflexibel und würde insbesondere die Gemeinsamkeit der verschiedenen Transaktionsmodelle übersehen. Benötigt wurde daher ein Beschreibungssystem, das die Gemeinsamkeiten verschiedener Transaktionsmodelle identifiziert und so die Schaffung eines Baukastens für die Bereitstellung unterschiedlicher Transaktionsmodelle bereitstellt.

Ein erster Ansatz hierzu ist das ACTA-Konzept [ChRa90], [KaRa95], das die Beschreibung unterschiedlicher Transaktionsmodelle durch Ereignisabhängigkeiten ermöglicht. Allerdings sind die in ACTA identifizierten Ereignisabhängigkeiten weder vollständig noch orthogonal. Daher wurde mit *AbstrACTA* ein vollständiges und orthogonales ereignisorientiertes Beschreibungssystem für Transaktionsmodelle geschaffen [Schm93], [KLSS94]. Dieses Beschreibungssystem wurde als Grundlage für die Gestaltung des Korrektheits- und Transaktionsaspekts herangezogen.

Jede der in AbstrACTA enthaltenen Ereignisabhängigkeiten wird in ein Aspektelement umgesetzt. Durch Kombination verschiedener Ereignisabhängigkeiten kann eine Transaktionsunterstützung erreicht werden, die die Nachbildung beliebiger Transaktionsmodelle erlaubt. Insbesondere konnte auf diese Weise eine Mehrfachimplementierung transaktionsunterstützender Funktionalität verhindert werden.

12.2.4 Bewertung

Im Rahmen des CompFlow-Projekts wurden alle Konzepte dieser Arbeit erfolgreich eingesetzt. So kamen das Zwei-Ebenen-Workflow-Modell, das Workflow-Modell-Wörterbuch, die aspektelementorientierte Schemarepräsentation und das inkrementelle Abbildungsverfahren zum Einsatz.

Zur Implementierung wurde die Enterprise JavaBeans Technologie verwendet. Da die EJB-Technologie das Konzept der kompositen Anwendung nicht kennt, wurde eine Umsetzung vorgestellt. Dazu mußte insbesondere ein erweiterter Spezialisierungsmechanismus geschaffen werden. Er erlaubt eine registraturbasierte Spezialisierung von EJB-Komponenten durch die Bildung von Komponentenuntertypen, während standardmäßig nur eine generelle Spezialisierung zur Verfügung steht. Auf dieser Grundlage wurde die Implementierung der aspektelementorientierten Schemarepräsentation in drei Schritten durchgeführt. Abschließend wurde die Implementierung des Beispielschemas mit Hilfe der EJB-Technologie beschrieben.

Ebenfalls erreicht wurde die Implementierung des Workflow-Modell-Wörterbuchs. Als Grundlage hierfür diente das Zwei-Ebenen-Workflow-Metamodell. Seine Einsetzbarkeit wurde bei der Einführung zweier komplett neuer Aspekte, nämlich des Korrektheits- und des Transaktionsaspekt geschildert. Diese zeigte, daß es nicht nur möglich ist einzelne Aspektelemente, sondern auch komplette Aspekte sowohl auf logischer als auch auf physischer Ebene neu aufzunehmen. Auf diese Weise konnte daher die Menge der unterstützen Workflow-Schemata vergrößert werden.

12.3 SOMFlow

12.3.1 Projektziel

Im Rahmen einer Industriekooperation wurde das Projekt SOMFlow [Scho98] durchgeführt. Ziel des Projektes war die Schaffung eines inkrementellen Abbildungsverfahrens für Ge-

schäftsprozeßschemata auf der Basis des SOM-Modells auf Workflow-Schemata des WfMS IBM FlowMark [Flow]. Die Herausforderung dieses Projektes lag darin, nicht mehr nur eine Gesamtabbildung, sondern eine inkrementelle Abbildung der Geschäftsprozeßschemata zu schaffen. Auf diese Weise sollte ohne erneute Gesamtabbildung die Einbringung von Änderungen der SOM-Schemata in die FlowMark Schemata möglich sein.

12.3.2 Lösungsansatz

Um dies zu erreichen, wurde das in Kapitel 11 entwickelte inkrementelle Abbildungsverfahren eingesetzt. Das Vorgehen ist in Abbildung 168 dargestellt. Zunächst wurde eine Aspektseparation des SOM-Modells durchgeführt (1), im Anschluß eine Feinstrukturierung mit Hilfe von Klassifizierungsstrukturen (2). Während das in dieser Arbeit verwendete Workflow-Modell aspektsepariert ist, ist dies beim Workflow-Modell von FlowMark nicht der Fall. Daher mußte eine Aspektseparation des FlowMark-Workflow-Modells durchgeführt werden (3), die die Grundlage für die Feinstrukturierung des FlowMark-Workflow-Modells legte (4). Nach diesen beiden Schritten wurde eine Äquivalenzsicht geschaffen (5). Damit lag sowohl für SOM als auch für Flow-Mark ein genügend feines Raster vor, auf dem Abbildungsoperatoren definiert werden konnten (6). Diese Abbildungsoperatoren konnten sogar bijektiv ausgeführt werden, wodurch gegenüber dem in CompFlow-Projekt entwickelten Verfahren ein weiterer Fortschritt erreicht werden konnte.

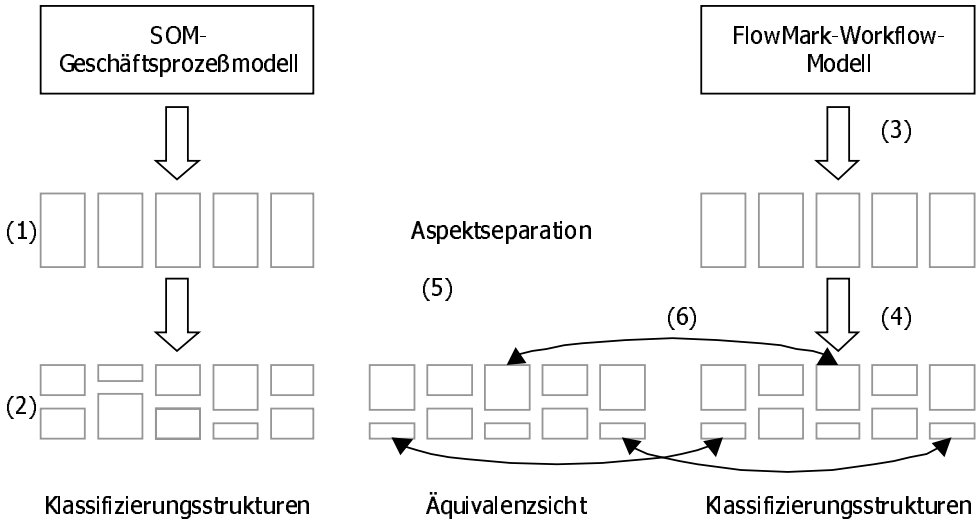


Abbildung 168: SOMFlow

Die identifizierten bijektiven Abbildungsoperatoren erlauben eine inkrementelle Abbildung von SOM-Geschäftsprozeßschemata auf FlowMark-Workflow-Schemata. Dargestellt ist dies in Abbildung 169. Dazu wird das SOM-Geschäftsprozeßschema mit Hilfe der Äquivalenzsicht in ein semantisch äquivalentes Schema umgewandelt (7). Auf dieses können dann die bijektiven Abbildungsoperatoren angewendet werden (8). Ergebnis ihrer Anwendung ist ein FlowMark-Workflow-Schema-Gerüst. Dieses Gerüst wird um Informationen ergänzt, die nicht im SOM-basierten Geschäftsprozeßschema enthalten sind (9). Ergebnis ist ein vollständiges Workflow-Schema für das WfMS FlowMark.

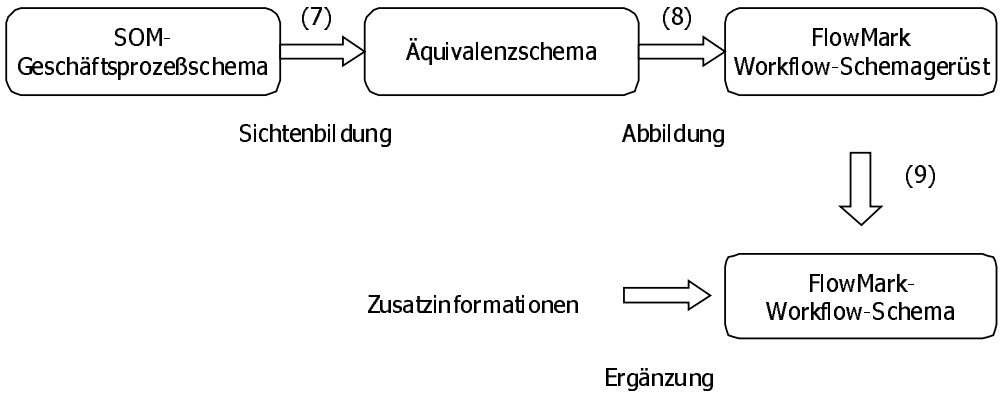


Abbildung 169: Anwendung der bijektiven Abbildungsoperatoren

Gegenüber dem in CompFlow entwickelten Abbildungsverfahrens konnte in SOMFlow durch die Schaffung der bijektiven Abbildungsoperatoren ein Fortschritt erzielt werden. So wurde nicht nur die inkrementelle Abbildung von Geschäftsprozeß- auf Workflow-Schemata, sondern sogar die sogenannte konzertierte Evolution [Scho98] von Geschäftsprozeß- und Workflow-Schemata erreicht. Hierunter versteht man, daß nicht nur Änderungen des Geschäftsprozeßschemas in das Workflow-Schema abgebildet werden, sondern auch umgekehrt Änderungen des Workflow-Schemas in Änderungen des Geschäftsprozeßschemas umgesetzt werden können.

12.3.3 Bewertung

Die Brauchbarkeit und Praxistauglichkeit des in dieser Arbeit entwickelten Konzepts für die inkrementelle Abbildung von Geschäftsprozeß- auf Workflow-Schemata wurde in diesem Projekt voll bestätigt. Ein zusätzlicher Fortschritt konnte durch die Schaffung von bijektiven Abbildungsoperatoren erreicht werden. Mit ihnen können auf Ebene der Workflow-Schemata durchgeführte Änderungen in das Geschäftsprozeßschema eingebracht werden. Auf diese Weise kann das in der Praxis oft zu beobachtende Auseinanderdriften von Geschäftsprozeß- und Workflow-Schemata noch besser verhindert werden als mit der inkrementellen Abbildung.

12.4 VORREITER

12.4.1 Projektziel

Das Projektziel des VORREITER-Projekts ist bereits in Kapitel 3 im Rahmen des Szenarios ausführlich vorgestellt worden. Es wird deswegen nur kurz rekapituliert. Im Rahmen des VORREITER-Projekts wird die Zusammenarbeit mehrerer klein- und mittelständischer Unternehmen (KMUs) im Rahmen von virtuellen Unternehmen durch eine geeignete informationstechnische Unterstützung ermöglicht. Die beteiligten Unternehmen stellen Hard- und Software für einzelne Bestandteile von Maschinensteuerungen her, können jeder für sich jedoch keine kompletten Maschinensteuerungen bereitstellen. Durch die Bildung eines virtuellen Unternehmens sollen die an VORREITER beteiligten Firmen in die Lage versetzt werden, auch als Anbieter für komplette Maschinensteuerungen aufzutreten.

Die Herausforderung liegt dabei in der informationstechnischen Unterstützung der weitreichenden Geschäftsprozesse innerhalb des virtuellen Unternehmens. Eine wichtige Randbedingung ist dabei, daß als komponentenorientiertes Rahmenwerk im VORREITER-Projekt die COM-Technologie [COM] verwendet wird. Daher soll zunächst eine kurze Darstellung dieser Technologie gegeben werden.

12.4.2 Die COM-Technologie

Zunächst soll gezeigt werden, wie in der COM-Technologie die Konzepte der Komponente und des komponentenorientierten Rahmenwerks umgesetzt werden. Dabei soll eine ähnliche Sprachregelung gelten wie bei der EJB-Technologie. Der Begriff COM soll verwendet werden, wenn Bestandteile der COM-Technologie gemeint sind, die das Konzept des komponentenorientierten Rahmenwerks unterstützen. Von COM-Komponenten wird die Rede sein, wenn es sich um Bestandteile der COM-Technologie handelt, die das Konzept der Komponente unterstützen.

12.4.2.1 Die COM-Technologie als komponentenorientiertes Rahmenwerk

In diesem Abschnitt wird gezeigt, wie Bestandteile der COM-Technologie das Konzept des komponentenorientierten Rahmenwerks umsetzen. Die Ergebnisse sind in Abbildung 170 dargestellt.

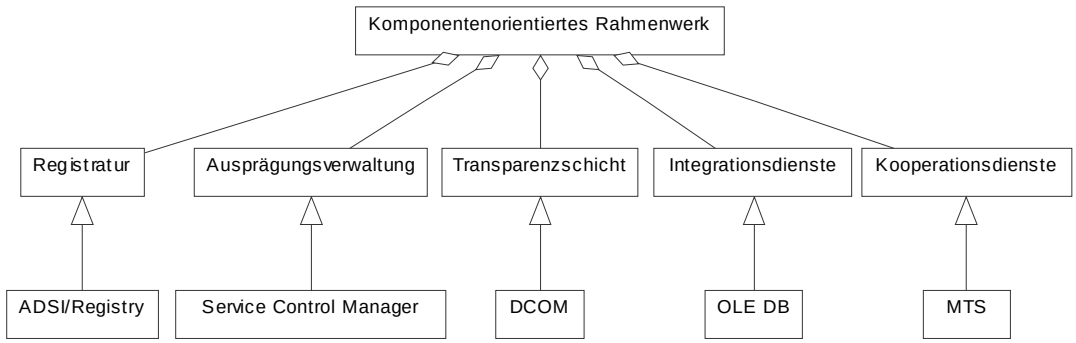


Abbildung 170: COM als komponentenorientiertes Rahmenwerk

Die Rolle der Registratur übernehmen zwei Bestandteile der COM-Technologie, die sogenannte Registry und der sogenannte Active Directory Service (ADSI). In der Registratur werden auch Spezialisierungsinformationen gespeichert, die bei der Bildung von Komponentenausprägungen zum Einsatz kommen. Der Ausprägungsverwaltung entspricht der sogenannte Service Control Manager. Mit ihm ist es möglich, transparent Ausprägungen von COM-Komponenten zu erzeugen. Die Rolle der Transparenzschicht übernimmt die Zusammenstellung eines Protokolls und mehrerer Mechanismen, die unter dem Namen Distributed COM (DCOM) zusammengefaßt werden. Die Integrationsdienste sind in Form von OLE-DB ausgeführt. Als Kooperationsdienst stellt COM den Microsoft Transaction Server (MTS) bereit. Mit ihm ist es möglich COM-Komponenten in Transaktionskontexten auszuführen.

Die Struktur der Registratur in der COM-Technologie soll näher betrachtet werden. Sie ist in Abbildung 171 dargestellt. Die Registratur, bestehend aus dem ADSI und der Registry, ist nach Komponententypen in Form sogenannter ClassIDs organisiert. Zu jeder dieser ClassIDs existiert genau ein Registry-Eintrag, der unter anderem den Ort der Komponente beschreibt. Daneben enthält der Registry-Eintrag auch Spezialisierungsinformationen für die COM-Komponente. Hieraus ergeben sich ähnliche Probleme wie bei der EJB-Technologie. Es ist nur eine generelle Spezialisierung möglich. Schnittstellenbeschreibungen werden in Form von Einträgen in die sogenannte Typelib repräsentiert.

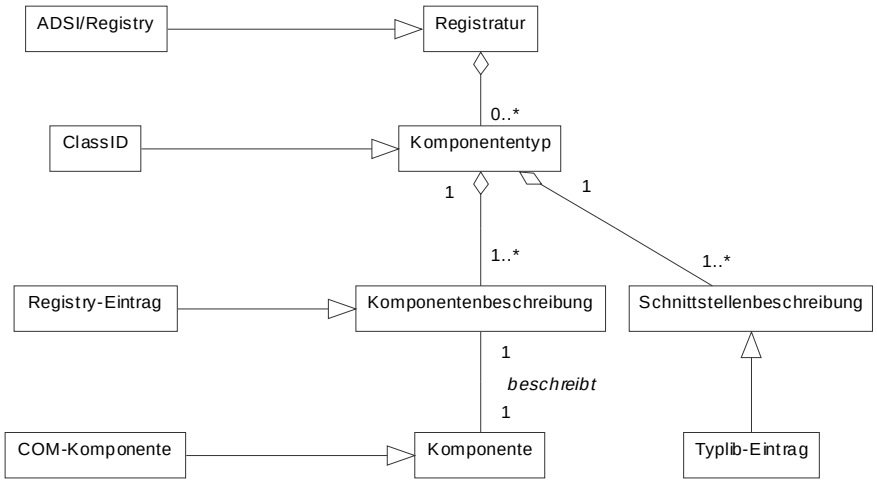


Abbildung 171: COM-Registratur

12.4.2.2 COM-Komponenten

Jetzt soll untersucht werden, in wieweit COM-Komponenten dem Konzept der Komponente entsprechen. Die Ergebnisse sind in Abbildung 172 dargestellt. Die Komponenten der COM-Technologie können eine oder mehrere Schnittstellen besitzen. Über die Struktur dieser Schnittstellen gibt die sogenannte Typelib Auskunft. Diese ist ein Teil der Introspektionsmechanismen. Der Registry-Eintrag des Komponententyps, unter dem die Komponente gespeichert ist, enthält Spezialisierungsinformationen, die bei der Bildung von Ausprägungen von COM-Komponenten verwendet werden. Hierdurch ist eine generelle Spezialisierung möglich.

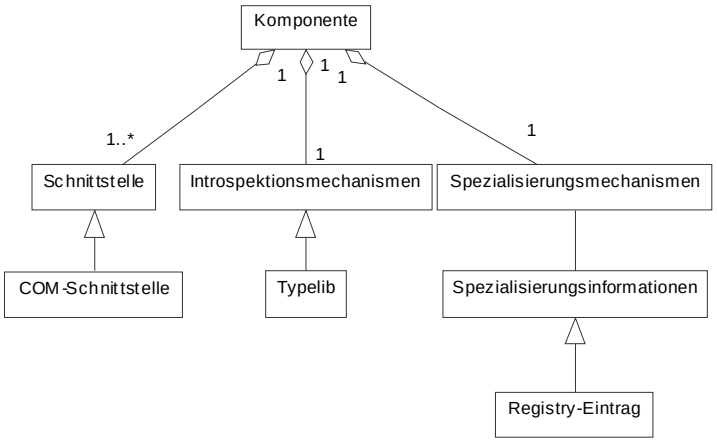


Abbildung 172: COM-Komponenten

12.4.2.3 Komposite Anwendungen

Ähnlich wie bei der EJB-Technologie ist auch der COM-Technologie das Konzept der kompositen Anwendung unbekannt. Deswegen muß im VORREITER-Projekt eine Umsetzung des Konzepts der kompositen Anwendung auf der Basis der COM-Technologie entwickelt werden, um ein vollständiges Komponentensystem zu erhalten.

Ausgangspunkt hierfür ist wiederum die Schaffung eines registraturbasierten Spezialisierungsmechanismusses, da mit COM-Komponenten nur eine generelle Spezialisierung möglich ist. Dafür wird ein ähnliches Verfahren wie bei der EJB-Technologie angewandt. Es werden zusätzliche ClassIDs zur Bildung der Komponentenuntertypen verwendet. Auf dieser Grundlage kann dann die Umsetzung der kompositen Anwendung durchgeführt werden. Dazu werden die Verknüpfungs- und Parametrisierungsinformationen in der Registry unter einem entsprechenden

Komponentenuntertyp abgelegt. Das Vorgehen gleicht dabei demjenigen, das im CompFlow-Projekt zur Bildung von kompositen Anwendungen auf Basis der EJB-Technologie angewandt wurde.

12.4.3 Lösungsansatz

Zur Unterstützung der weitreichenden Geschäftsprozesse im VORREITER-Projekt werden alle in dieser Arbeit entwickelten Konzepte eingesetzt. Durch die Verwendung des in Kapitel 6 entwickelten Zwei-Ebenen-Workflow-Metamodells wird die Autarkie der beteiligten Unternehmen bezüglich ihrer informationstechnischen Unterstützung gesichert. Auf diese Weise können die am VORREITER-Projekt beteiligten Unternehmen informationstechnische Ressourcen eigenständig zuordnen.

Um Erweiterungen des Workflow-Modell durchführen zu können, wird im VORREITER-Projekt eine dynamische Repräsentation des Workflow-Modells durch ein Workflow-Modell-Wörterbuch vorgenommen. Das Workflow-Modell-Wörterbuch im VORREITER-Projekt setzt sich aus einzelnen Workflow-Modell-Wörterbüchern zusammen, die sich bei den Teilnehmern des virtuellen Unternehmens befinden. Veranschaulicht ist dies in Abbildung 173.

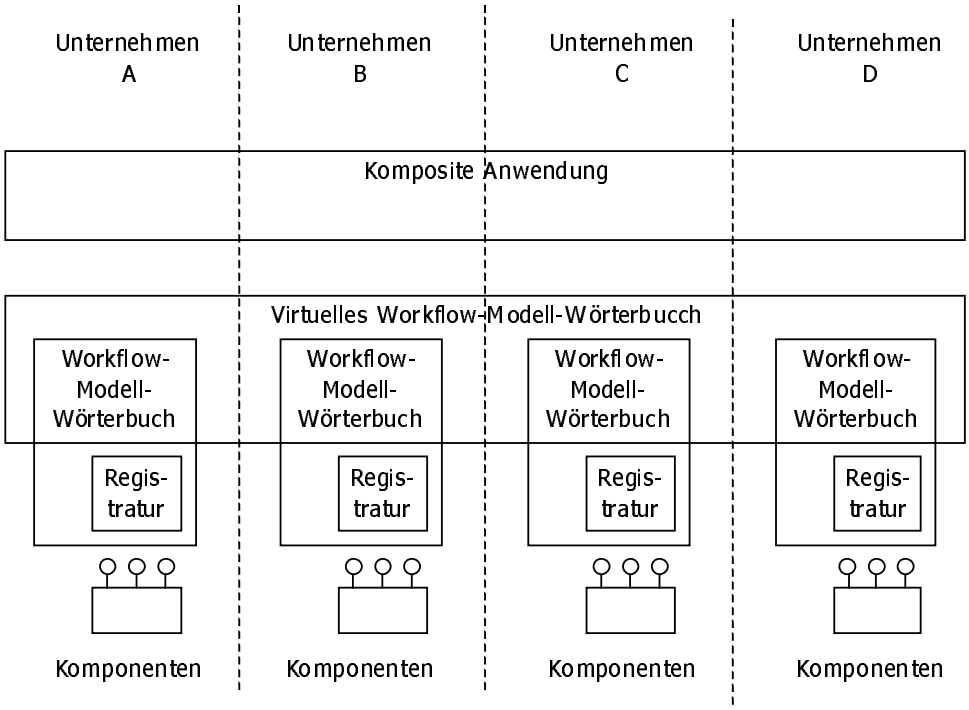


Abbildung 173: VORREITER-Projekt

Zusammen bilden sie ein virtuelles Gesamtwörterbuch, das eine transparente Nutzung aller Bestandteile des Workflow-Modells ermöglicht. Dazu wird die Idee aus Abschnitt 9.3.1.3 aufgegriffen, die Spezialisierungsinformationen auf mehrere Registraturen zu verteilen. In dieses Wörterbuch sind die Registraturen der komponentenorientierten Rahmenwerke integriert, hier also die Registratur auf der Basis der COM-Technologie in Form der Registry und des Active Directory Service (ADSI).

Im VORREITER-Projekt wird die aspektelementorientierte Schemarepräsentation auf der Basis kompositer Anwendungen erstellt, die durch COM-Komponenten gebildet wird. Die kompositen Anwendungen erstrecken sich dabei über das gesamte virtuelle Unternehmen. Die dafür verwendeten Komponenten sind über die Projektpartner verteilt und in deren Registraturen eingetragen. Die Projektpartner können diese Komponenten auswechseln, ohne daß die übrigen

Partner hiervon betroffen wären. Ebenso ist eine Anpassung von Diensten möglich, ohne daß andere Partner davon beeinflußt würden. Auf diese Weise kann die Autarkie der Prozeßteilnehmer in vollem Umfang gewährleistet werden.

Das inkrementelle Abbildungsverfahren wird in CompFlow eingesetzt, um Änderungen der weitreichenden Geschäftsprozesse mit minimalem Aufwand einbringen zu können. Dies ist besonders durch die Vielzahl eigenständiger Teilnehmer bei virtuellen Unternehmen von Vorteil, da jede Änderung mit dem betroffenen Teilnehmer abgestimmt werden muß. Ein inkrementelles Abbildungsverfahren hilft daher sehr die Anzahl der Abstimmungsvorgänge zu minimieren.

12.4.4 Bewertung

Zwar befindet sich das VORREITER-Projekt noch in der Konzeptionsphase, schon jetzt kann aber festgestellt werden, daß die in dieser Arbeit entwickelten Konzepte für die Bedürfnisse dieses Projekts voll tauglich sind. Auch die Anwendung der in dieser Arbeit entwickelten Konzepte auf COM stellt kein Problem dar, wie die Einordnung der Bestandteile der COM-Technologie in die Konzepte des komponentenorientierten Rahmenwerks und der Komponente gezeigt hat.

12.5 Zusammenfassung

In diesem Kapitel wurde die Implementierung der in dieser Arbeit entwickelten Konzepte im Rahmen mehrerer Projekte geschildert.

Im WOGE-Projekt wurde die aspektelementorientierte Schemarepräsentation und das Zwei-Ebenen-Workflow-Modell erfolgreich umgesetzt, was nicht zuletzt durch den operativen Einsatz der Projektergebnisse beim Projektpartner bestätigt wird. Im CompFlow Projekt kamen alle in dieser Arbeit entwickelten Konzepte erfolgreich zum Einsatz. Als technische Grundlage wurde dabei die Enterprise JavaBeans Technologie verwendet. Im SOMFlow Projekte wurde das inkrementelle Abbildungsverfahren für die Abbildung von SOM-Geschäftsprozeßschemata auf FlowMark-Workflow-Schemata verwendet. Dabei konnte sogar eine konzertierte Evolution beider Schemata sichergestellt werden. Das VORREITER Projekt bringt ebenfalls alle Konzepte dieser Arbeit zum Einsatz. Der Einsatz in unterschiedlichen Projekten zeigte weiterhin, daß die entwickelten Konzepte leicht auf unterschiedliche Technologien wie beispielsweise COM oder auch Lotus Notes übertragen werden können.

13 Evaluierung

In diesem Kapitel wird überprüft, ob das Ziel der Arbeit, nämlich die informationstechnische Unterstützung weitreichender Prozesse, erreicht wird. Kriterium hierfür sind die in Kapitel 3 identifizierten Anforderungen an eine informationstechnische Unterstützung weitreichender Prozesse. So muß die Autarkie bei der informationstechnischen Umsetzung sichergestellt sein, d.h. es muß eine feingranulare und dynamische Zuweisung informationstechnischer Ressourcen möglich sein. Die Menge der unterstützten Workflow-Schemaelemente darf zudem nicht durch ein a priori festgelegtes Workflow-Modell eingeschränkt werden. Statt dessen muß es möglich sein, Erweiterungen des Workflow-Modells der informationstechnischen Unterstützung durchführen zu können. Die Änderungen der Geschäftsprozesse müssen inkrementell auf Workflow-Schemata abgebildet werden. Diese veränderten Workflow-Schemata müssen flexibel umgesetzt werden.

Damit ein Ansatz zur Unterstützung weitreichender Prozesse in der Lage ist, müssen diese Anforderungen in ihrer Gesamtheit erfüllt werden. Kein existierender Ansatz war hierzu in der Lage, wie Kapitel 4 gezeigt hat. In dieser Arbeit wurde daher eine Reihe von Konzepten entwickelt. Es sind dies das Zwei-Ebenen-Workflow-Metamodell, das Workflow-Modell-Wörterbuch, die aspektelementorientierte Schemarepräsentation und das inkrementelle Abbildungsverfahren für Geschäftsprozeßschemata. Um zu zeigen, daß die Erfüllung der Anforderungen in ihrer Gesamtheit durch die in dieser Arbeit entwickelten Konzepte möglich ist, wird für jede der Anforderungen der Nachweis geführt, daß sie erfüllt wird.

Das Vorgehen dazu ist folgendes: Zunächst wird die Herleitung der Anforderung aus den Rahmenbedingungen für die Unterstützung weitreichender Prozesse rekapituliert. Danach wird auf die Erfahrungen aus dem praktischen Einsatz der Lösungskonzepte dieser Arbeit zurückgegriffen, die im vorangegangenen Kapitel beschrieben worden sind. Mit ihnen wird überprüft, ob die Anforderung im Rahmen von Projekten auch erfüllt werden konnte. In Tabelle 14 ist dargestellt, in welchen Projekten die in dieser Arbeit entwickelten Konzepte eingesetzt wurden.

	Zwei-Ebenen-Workflow-Metamodell	Workflow-Modell-Wörterbuch	Aspektelementorientierte Schemarepräsentation	Inkrementelle Abbildung von Geschäfts-prozeß- auf Workflow-Schemata
WOGE	+		+	
CompFlow	+	+	+	+
SOMFlow				+
VORREITER	+	+	+	+

Tabelle 14: Projekteinsatz der Lösungskonzepte

Auf der Grundlage der Erfahrungen aus dem praktischen Einsatz der Projekte werden die Gründe für ihren erfolgreichen Einsatz näher dargestellt. Dabei wird die Frage geklärt werden, wodurch die Erfüllung einer Anforderung durch ein Lösungskonzept herrührt. Um den erzielten Neuigkeitswert zu verdeutlichen, wird dann gezeigt, wieso die existierenden Ansätze zur Erfüllung der Anforderung nicht in der Lage sind.

13.1 Autarkie bei der informationstechnischen Umsetzung

Die Anforderung der Autarkie der informationstechnischen Umsetzung ergibt sich aus der organisatorischen Autarkie der Prozeßteilnehmer. Die Prozeßteilnehmer wollen oder können sich nicht mehr einer zentralen Organisation unterwerfen. Daher kann nicht mehr davon ausgegangen werden, daß sich die Teilnehmer des weitreichenden Prozesses die Ressourcen zur informationstechnischen Unterstützung und deren Ausgestaltung vorgeben lassen. Die Prozeßteilnehmer möchten für einzelne Prozeßschritte informationstechnische Ressourcen zuweisen können. Die Zuweisung informationstechnischer Ressourcen muß feingranular erfolgen können. Durch die Autarkie der Prozeßteilnehmer kommt es zudem zu einer unabhängigen Evolution der informationstechnischen Ressourcen. Um dem gerecht zu werden, muß die Zuweisung informationstechnischer Ressourcen zur Laufzeit möglich sein.

13.1.1 Praktische Evaluierung

Das in dieser Arbeit entwickelte Zwei-Ebenen-Workflow-Metamodell ermöglicht erstmalig die dynamische und feingranulare Zuweisung von informationstechnischen Ressourcen und sichert so die Autarkie bei der informationstechnischen Umsetzung. Dies bestätigte sich durch seinen Einsatz in mehreren Projekten.

So wurde im WOGE-Projekt, die Autarkie bei der informationstechnischen Umsetzung des Geschäftsprozesses CSPA durch die Verwendung des Zwei-Ebenen-Workflow-Metamodells als Grundlage für das Workflow-Modell erreicht. Während im WOGE-Projekt das Workflow-Modell noch statisch repräsentiert wurde, wurde im CompFlow-Projekt durch die Verwendung eines Workflow-Modell-Wörterbuchs eine dynamisch anpaßbare Repräsentation erreicht. Daher konnte nicht nur die Autarkie bei der informationstechnischen Unterstützung von Geschäftsprozessen sichergestellt werden, sondern auch die Erweiterbarkeit des Workflow-Modells. Im VORREITER-Projekt wird dies noch weiter ausgeführt, indem das Zwei-Ebenen-Workflow-

Metamodell die Grundlage für Workflow-Modell-Wörterbücher bildet, die zusammen ein virtuelles Gesamtwörterbuch bilden.

13.1.2 Lösungskonzept: Zwei-Ebenen-Workflow-Metamodell

Die Verwendung von zwei Ebenen im Workflow-Metamodell ist die Grundlage für die flexible Zuweisung informationstechnischer Ressourcen. Die logische Ebene legt die Grundlage für eine von konkreten informationstechnischen Ressourcen unabhängige Darstellung von Workflow-Schemata. In der physischen Ebene befindet sich zu jedem Aspekt der logischen Ebene ein korrespondierender Aspekt, der die informationstechnischen Ressourcen zur Unterstützung des logischen Aspekts darstellt. Die Aspekte unterteilen sich weiter in Aspektelemente. Das Zwei-Ebenen-Workflow-Metamodell ist in Abbildung 174 dargestellt.

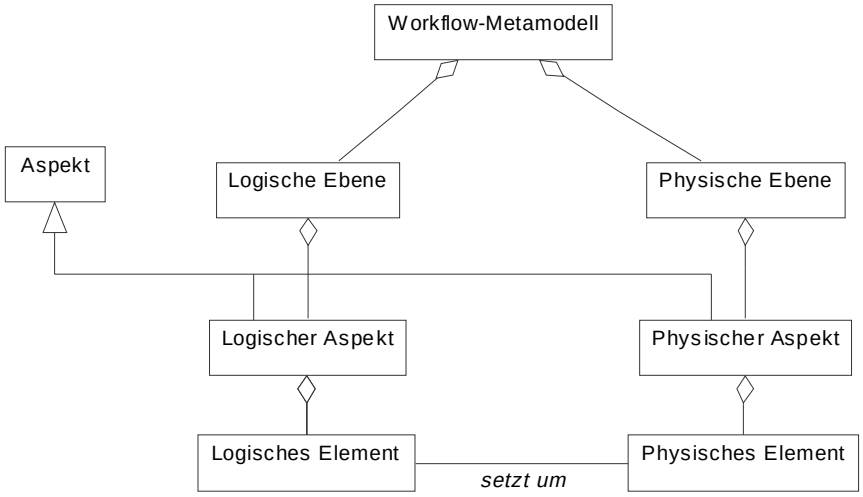


Abbildung 174: Zwei-Ebenen-Workflow-Metamodell

Um eine feingranulare Zuweisung zu ermöglichen, geschieht die Zuordnung zwischen logischer und physischer Ebene nicht mit ganzen Aspekten als Granulat, sondern Aspektelementen. Daher können für einzelne Elemente eines Aspektes unterschiedliche informationstechnische Ressourcen zur Unterstützung zugewiesen werden. Jedes Element eines physischen Aspekts wird durch ein oder mehrere Dienste unterstützt, die wiederum durch mehrere unterschiedliche Ressourcen erbracht werden können. Die Zergliederung in Dienste und Ressourcen ermöglicht es, Dienste zu erfassen, die bei der Unterstützung mehrerer Aspektelemente beteiligt sind, sowie Ressourcen darzustellen, die mehrere Dienste bzw. Aspektelemente unterstützen. Veranschaulicht ist dies in Abbildung 175.

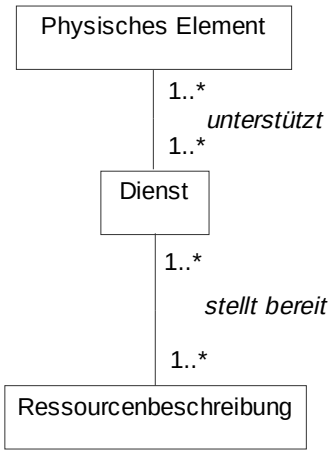


Abbildung 175: Dienste und Ressourcen

Durch die Möglichkeit die Dienste erbringenden Ressourcen beliebig zu gruppieren, eröffnet sich ein weites Optimierungspotential. Für eine maximale Flexibilität bei der Ressourcenzuordnung kann es sinnvoll sein, jeden Dienst durch eine eigene Ressource bereitzustellen. Andererseits sind auch Fälle denkbar, bei denen es günstiger ist, bestimmte Menge von Diensten durch dieselbe Ressource erbringen zu lassen. Dies ist beispielsweise der Fall, wenn die Ressourcen in hohem Maße redundante Funktionalität aufweisen.

13.1.3 Existierende Ansätze

Die autarke Zuweisung von Ressourcen ist bei den existierenden Ansätzen bereits durch die Gestaltung des Workflow-Metamodells versperrt. Es findet keine Trennung zwischen einer logischen und einer physischen Ebene statt. So erlauben es die existierenden Ansätze nicht, die zur Umsetzung von Prozessen verwendeten Ressourcen zu beschreiben und eine flexible Zuweisung durchzuführen.

13.2 Erweiterbarkeit des Workflow-Modells

Bei weitreichenden Prozessen ist mit tiefer greifenden Änderungen der Prozeßschemata und damit auch der Workflow-Schemata zu rechnen als bisher. Die Zahl der Teilnehmer ist größer und dementsprechend auch die Zahl von potentiellen Quellen für Änderungen. Daher muß es möglich sein, die Menge der unterstützten Workflow-Schemaelemente, also das Workflow-Modell, zu erweitern.

13.2.1 Praktische Evaluierung

Das in dieser Arbeit geschaffene Workflow-Modell-Wörterbuch ermöglicht es, das Workflow-Modell zur Laufzeit zu erweitern. Auf diese Weise können Prozeßänderungen unterstützt werden, die eine Erweiterung des Workflow-Modells erforderlich machen. Bestätigt wurde dies durch den erfolgreichen Einsatz des Workflow-Modell-Wörterbuchs in mehreren Projekten.

Das Workflow-Modell-Wörterbuchs wurde zuerst im CompFlow-Projekt eingesetzt. Besondere Aufmerksamkeit wurde dabei den bereits in der EJB-Registratur vorliegenden Informationen geschenkt. Um deren Nutzung zu ermöglichen, wurde ein Integrationsmodell entwickelt. Im CompFlow-Projekt konnte nicht nur die Aufnahme zusätzlicher Aspektelemente, sondern sogar die Aufnahme zweier neuer Aspekte, nämlich des Korrektheits- und des Transaktionsaspekts, praktisch umgesetzt werden. Im VORREITER-Projekt wurde für den Einsatz in virtuellen Un-

ternehmen eine Aufteilung des Workflow-Modell-Wörterbuchs und der Registraturen in mehrere Teilwörterbücher und –registraturen konzipiert. Zur informationstechnischen Prozeßunterstützung werden diese Wörterbücher und Registraturen wiederum zu einem virtuellen Gesamtwörterbuch zusammengeführt.

13.2.2 Lösungskonzept: Workflow-Modell-Wörterbuch

Grundlage für die Erweiterbarkeit des Workflow-Modells ist die geeignete Gestaltung des Informationsmodells des Workflow-Modell-Wörterbuchs. Das Informationsmodell besteht aus sechs Objekttypen: Logische und physische Aspekte, logische und physische Elemente, Dienste und Ressourcenbeschreibungen. Die Darstellung des ersten Teils des Informationsmodells befindet sich in Abbildung 176. Es sind die Objekttypen „Logischer Aspekt“, „Logisches Element“, „Physischer Aspekt“ und „Physisches Element“. Zwischen den Objekttypen gibt es drei Beziehungen: die „Zugehörigkeit logischer Aspekt“, die Beziehung „Zugehörigkeit physischer Aspekt“ und die Umsetzungsbeziehung.

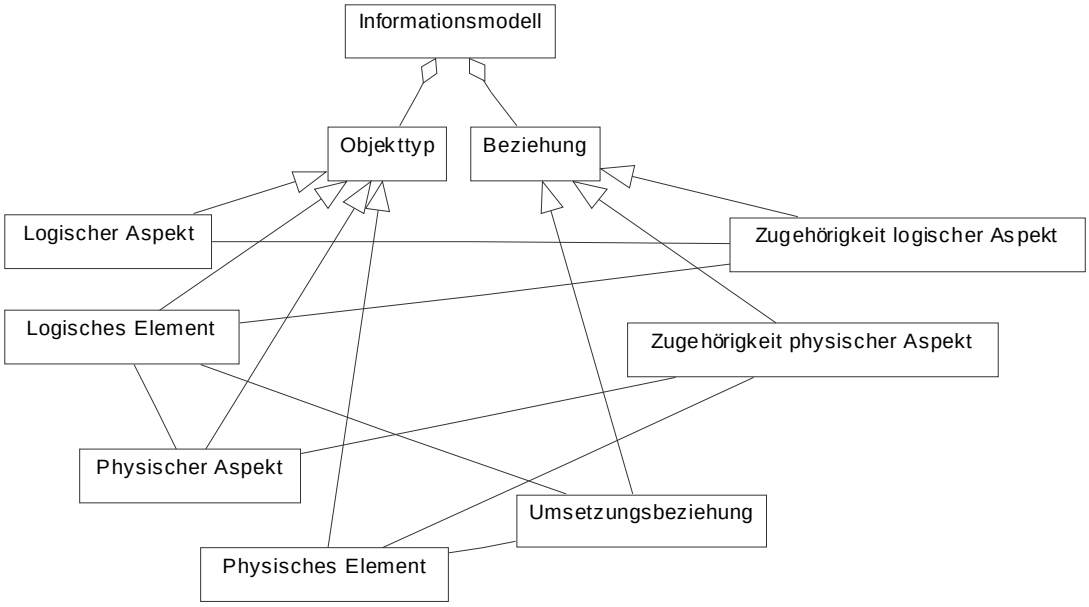


Abbildung 176: Informationsmodell des Workflow-Modell-Wörterbuchs (1)

Der zweite Teil des Informationsmodells, der in Abbildung 177 wiedergegeben ist, besteht aus den Objekttypen „Physisches Element“, „Dienstbeschreibung“ und „Ressourcenbeschreibung“. Zwischen den Objekttypen bestehen zwei Beziehungen: die Unterstützungsbeziehung und die Bereitstellungsbeziehung.

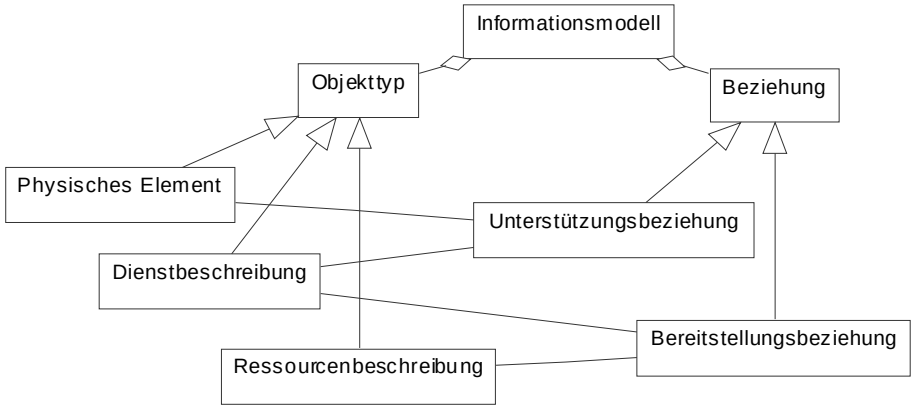


Abbildung 177: Informationsmodell des Workflow-Modell-Wörterbuchs (2)

Das Workflow-Modell-Wörterbuch kann sowohl zusätzliche logische und physische Elemente aufnehmen, als auch Dienste und Ressourcen. Dies wird durch Operationen auf den im Workflow-Modell-Wörterbuch befindlichen Informationen umgesetzt.

13.2.3 Existierende Ansätze

Die Anforderung der dynamischen Erweiterbarkeit kann von den bisherigen Ansätzen nicht erfüllt werden, da ihnen bis auf eine Ausnahme ein a priori bestimmtes Workflow-Modell fest eingeprägt ist. So verwenden die existierenden Ansätze zwar Wörterbücher für die Repräsentation von Workflow-Schemata, nicht jedoch für die Repräsentation des Workflow-Modells. Daher können sie zur Laufzeit zusätzliche Schemata aufnehmen, die dafür zur Verfügung stehende Menge von Schemaelemente, sprich das Workflow-Modell, ist jedoch nicht erweiterbar.

Der erzielte Fortschritt kann an Hand der Darstellung der Phasen in einem WfMS verdeutlicht werden, die in Kapitel 2 eingeführt worden sind. In Abbildung 178 ist dargestellt, wie sich die Zeiträume für das Einbringen zusätzlicher Workflow-Modellelemente zwischen den existierenden Ansätzen und dem in dieser Arbeit entwickelten Ansatz unterscheiden. Während die existierenden Ansätze mit der Ausnahme des Mentor-lite-Ansatzes nur in der WfMS-Entwurfsphase Erweiterungen des Workflow-Modells zulassen, sind mit dem in dieser Arbeit entwickelten Workflow-Modell-Wörterbuch Erweiterungen in allen Phasen möglich.

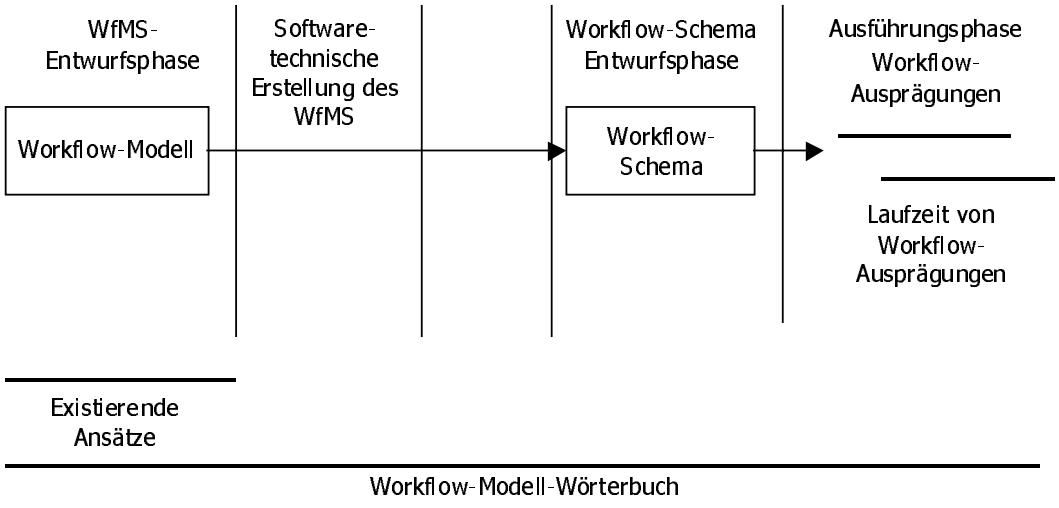


Abbildung 178: Phasen mit der Möglichkeit zur Erweiterung des Workflow-Modells

13.3 Flexibilität und Skalierbarkeit

Wegen der Häufigkeit von Prozeßänderungen und dementsprechenden Veränderungen der Workflow-Schemata ist es bei der Unterstützung von weitreichenden Prozessen notwendig, die Änderungen von Workflow-Schemata zur Laufzeit umsetzen zu können. Diese Anforderung wird als Flexibilität bezeichnet. Ebenso ist es erforderlich, daß nicht mehr nur eine Auswahl, sondern die Gesamtheit der Prozesse eines Unternehmens unterstützt wird. Hieraus ergibt sich die Anforderung der Skalierbarkeit der informationstechnischen Prozeßunterstützung.

13.3.1 Praktische Evaluierung

Durch die in dieser Arbeit konzipierte aspektelementorientierte Schemarepräsentation ist es erstmalig möglich, die Anforderungen der Flexibilität und Skalierbarkeit gleichzeitig zu erfüllen. Dies wurde durch den Einsatz in mehreren Projekten bestätigt.

Die aspektelementorientierte Schemarepräsentation wurde im WOGE-Projekt implementiert. Durch ihre Anwendung konnte die flexible Umsetzung von Workflow-Schemaänderungen erreicht werden. Allerdings bildete kein Komponentensystem, sondern Lotus Notes die Grundlage, so daß das Konzept der kompositen Anwendung nur annähernd verwirklicht werden konnte. Im CompFlow-Projekt konnte hingegen eine vollständige Implementierung sowohl der aspektelementorientierten Schemarepräsentation als auch der kompositen Anwendung durchgeführt werden, da die Enterprise JavaBeans Technologie als Grundlage verwendet wurde. Die dabei erhaltene Implementierung erwies sich als in hohem Maße anpassungsfähig und verträglich zu den übrigen Konzepten.

Im VORREITER-Projekt erstreckt sich die aspektelementorientierte Schemarepräsentation über mehrere Unternehmen, ohne aber deren Autarkie bezüglich der informationstechnischen Ressourcen einzuschränken. Die Schemarepräsentation wird durch eine komposite Anwendung umgesetzt, die von den beteiligten Unternehmen durch selbst zu bestimmende Komponenten unterstützt wird. Die Menge der Komponenten kann zudem erweitert werden und so auch die Menge der unterstützten Workflow-Schemaelemente.

Der durch die aspektelementorientierte Schemarepräsentation erreichte Fortschritt hat tiefgehende Konsequenzen für die informationstechnische Unterstützung weitreichender Prozesse. Bisher konnte durch die direkte Bildung von Ausprägungen zwar eine hohe Flexibilität erreicht werden, nicht jedoch auch eine hohe Skalierbarkeit, da die zentralisierte Interpretation des Workflow-Schemas einen Engpaß darstellte. Umgekehrt erreicht die indirekte Bildung von Ausprägungen eine hohe Skalierbarkeit, allerdings um den Preis der Flexibilität. Durch die aspektelementorientierte Schemarepräsentation ist es möglich, die Vorteile der direkten und der indirekten Bildung von Ausprägungen zu verbinden und so die Ziele der Skalierbarkeit und Flexibilität gleichzeitig zu erreichen.

Ein weiterer wichtiger Vorteil ist der verringerte Verbrauch technischer Ressourcen, wie beispielsweise Speicherplatz, durch die Verwendung der aspektelementorientierten Schemarepräsentation. Bei den bisherigen Ansätzen wurde eine Vielzahl von Diensten bei der Workflow-Ausführung aktiviert, auch wenn diese nicht benötigt wurden. Die aspektelementorientierte Schemarepräsentation erreicht durch die feingranulare Gestaltung bereits eine Reduzierung der gleichzeitig benötigten Dienste und damit der benötigten Ressourcen. Weiter reduziert wird der Ressourcenbedarf durch eine Kombination mit den Möglichkeiten der Ressourcenverwaltung, die durch Komponentensysteme bereitgestellt werden, wie in Abschnitt 9.3.2 beschrieben.

13.3.2 Lösungskonzept: Aspektelementorientierte Schemarepräsentation

Die aspektelementorientierte Schemarepräsentation ist Teil eines indirekten Ansatzes zur Bildung von Ausprägungen, wie in Abbildung 179 veranschaulicht. Daher wird die Anforderung der Skalierbarkeit erfüllt, wie die Diskussion in Abschnitt 5.1.2 gezeigt hat. Im Gegensatz zu den bisherigen Ansätzen erlaubt die aspektelementorientierte Schemarepräsentation jedoch eine schnelle Umsetzung von Schemaänderungen wodurch sie auch die Anforderung der Flexibilität erfüllt. Durch sie ist es daher möglich, die Vorteile der direkten und der indirekten Bildung von Ausprägungen zu verbinden und so die Ziele der Skalierbarkeit und Flexibilität gleichzeitig zu erreichen.

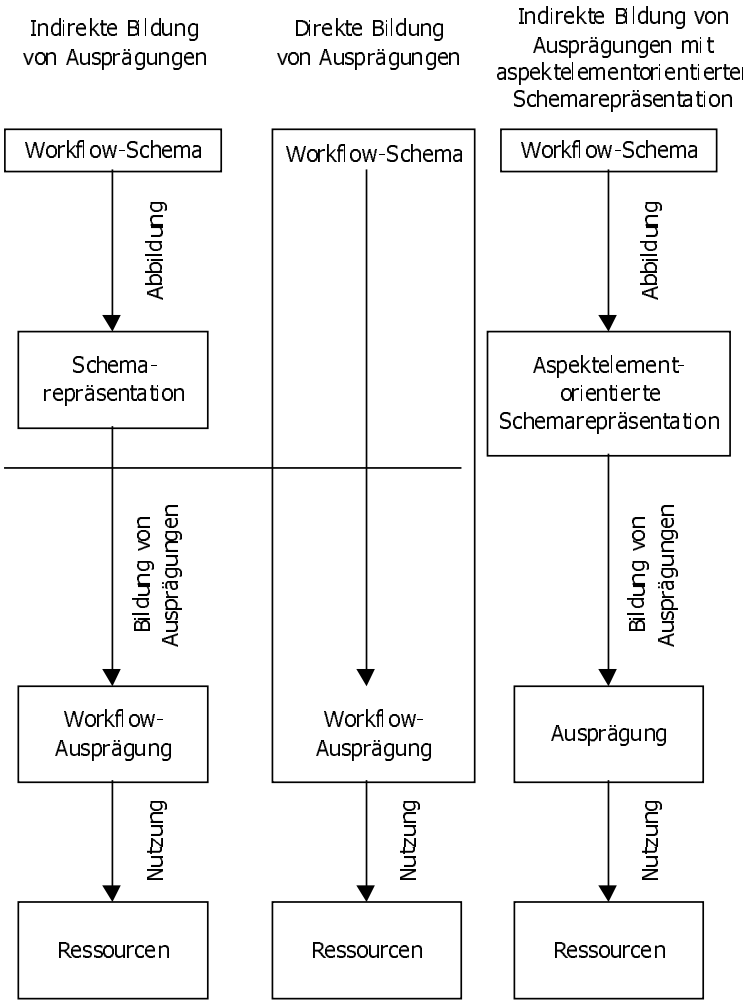


Abbildung 179: Indirekte und direkte Bildung von Ausprägungen

Die aspektelementorientierte Schemarepräsentation erreicht die schnelle Anpaßbarkeit an Schemaänderungen, indem sie Schemata als eine Menge von Repräsentationselementen mit Verbindungspunkten und Verbindern repräsentiert. Jedes der Repräsentationselemente gibt jeweils nur ein Aspektelement einer Schemarepräsentation wieder. Die Repräsentationselemente sind über Verbinder miteinander verbunden. Die Verbinder geben entweder temporale oder nicht-temporale Abhängigkeiten zwischen den Repräsentationselementen wieder. Durch diese Aufteilung lassen sich alle vier Arten von Schemaänderungen, also Schemaerweiterungen und –reduktionen, sowie Topologie- und Parameteränderungen unterstützen.

13.3.3 Existierende Ansätze

Die existierenden Ansätze lassen danach unterscheiden, ob sie Ausprägungen direkt oder indirekt bilden, wie in Abbildung 179 dargestellt. Beide Gruppen erfüllen entweder die Anforderung der Flexibilität oder der Skalierbarkeit. Kein Ansatz erfüllt jedoch beide Anforderungen gleichzeitig. Direkt Ausprägungen bildende Ansätze erreichen eine hohe Flexibilität bei eingeschränkter Skalierbarkeit. Ursache für die mangelnde Skalierbarkeit dieser Ansätze ist die zentralisierte Abbildung des Workflow-Schemas und die Zusammenfassung mit der Bildung von Ausprägungen. Indirekt Ausprägungen bildende Ansätze erreichen eine hohe Skalierbarkeit bei eingeschränkter Flexibilität. Die mangelnde Flexibilität dieser Ansätze ist durch die nicht anpassungsfähige Gestaltung der Schemarepräsentation begründet.

13.4 Inkrementelle Abbildung von Prozeßschemata auf Workflow-Schemata

Bei weitreichenden Prozessen existieren durch die Vielzahl und Heterogenität der Teilnehmer deutlich mehr Quellen für Änderungen an den Prozessen als bei Prozessen, die auf Abteilungsebene beschränkt sind. Auch die häufige zeitliche Begrenztheit von weitreichenden Prozessen, wie sie beispielsweise in virtuellen Unternehmen gegeben sind, erfordert, die zu ihrer Bildung notwendigen Anpassungen schnell durchführen zu können. Hieraus ergibt sich als eine Anforderung, Änderungen von Geschäftsprozeßschemata inkrementell auf Workflow-Schemata abbilden zu können.

13.4.1 Praktische Evaluierung

Das in diese Arbeit entwickelte Abbildungsverfahren ermöglicht im Gegensatz zu den existierenden Verfahren die inkrementelle Abbildung von Geschäftsprozeß- auf Workflow-Schemata. Die praktischen Erfahrungen aus mehreren Projekten bestätigen dies.

Im CompFlow-Projekt wurde das Verfahren für die Abbildung von Geschäftsprozeßschemata des SOM-Modells auf Workflow-Schemata erfolgreich eingesetzt. Eine Verfeinerung erfuhr das Verfahren im SOMFlow-Projekt. Hier konnte durch die Definition bijektiver Abbildungsoperatoren sogar eine konzertierte Evolution von Geschäftsprozeß- und Workflow-Schemata erreicht werden. Hierunter versteht man den wechselseitigen Abgleich von Änderungen in beiden Schemata. Es können also nicht nur Änderungen des Geschäftsprozeßschemas auf das Workflow-Schema, sondern auch umgekehrt Änderungen des Workflow-Schemas zum Geschäftsprozeßschema durchgereicht werden. Dadurch wurde der häufig zu beobachtende Effekt des Auseinanderdriftens von Geschäftsprozeß- und Workflow-Schema verhindert.

Durch das inkrementelle Abbildungsverfahren konnte in beiden Projekten der Aufwand zur Umsetzung von Änderungen eines Geschäftsprozeßschemas in ein verändertes Workflow-Schema drastisch reduziert werden. Besonders wichtig ist dies für virtuelle Unternehmen, wie sie im VORREITER-Projekt unterstützt werden. Hier erfordern Änderungen durch die Notwendigkeit, diese unter den Teilnehmern abzustimmen, einen besonders hohen Aufwand. Aus diesem Grund ist das inkrementelle Verfahren zur Abbildung von Geschäftsprozeß- auf Workflow-Schemata von großer Bedeutung für die Prozeßunterstützung in virtuellen Unternehmen.

13.4.2 Lösungskonzept: Inkrementelles Abbildungsverfahren

Das entwickelte Verfahren erreicht die Fähigkeit zur inkrementellen Abbildung, indem ein Raster entwickelt wird, das die vollständige und feingranulare Darstellung von Geschäftsprozeß- und Workflow-Modellen ermöglicht. Auf diese Weise werden minimale Änderungssphären ge-

schaffen, die als Ausgangspunkt und Ziel für Abbildungsoperatoren auf Modellebene dienen. Diese werden dann herangezogen, um die inkrementelle Abbildung der Geschäftsprozeß- auf die Workflow-Schemata durchzuführen. Dieses Vorgehen ist in Abbildung 180 veranschaulicht.

Der erste Schritt zur Schaffung des Rasters ist die Aspektseparation von Geschäftsprozeß- und Workflow-Modell (1). Dadurch werden die Auswirkungen von Änderungen auf Aspekte begrenzt. Im nächsten Schritt wird eine Feinzerlegung durch sogenannte Klassifizierungsstrukturen durchgeführt, mit denen das Raster weiter eingegrenzt wird (2). Der dritte Schritt ist schließlich die Schaffung einer Äquivalenzsicht für das Geschäftsprozeßmodell (3). Diese ist semantisch zum Geschäftsprozeßmodell und strukturell zum Workflow-Modell kompatibel. Zwischen der Äquivalenzsicht und dem Workflow-Modell werden schließlich die Operatoren definiert (4), die zur Abbildung vom Geschäftsprozeß- auf das Workflow-Schema verwendet werden.

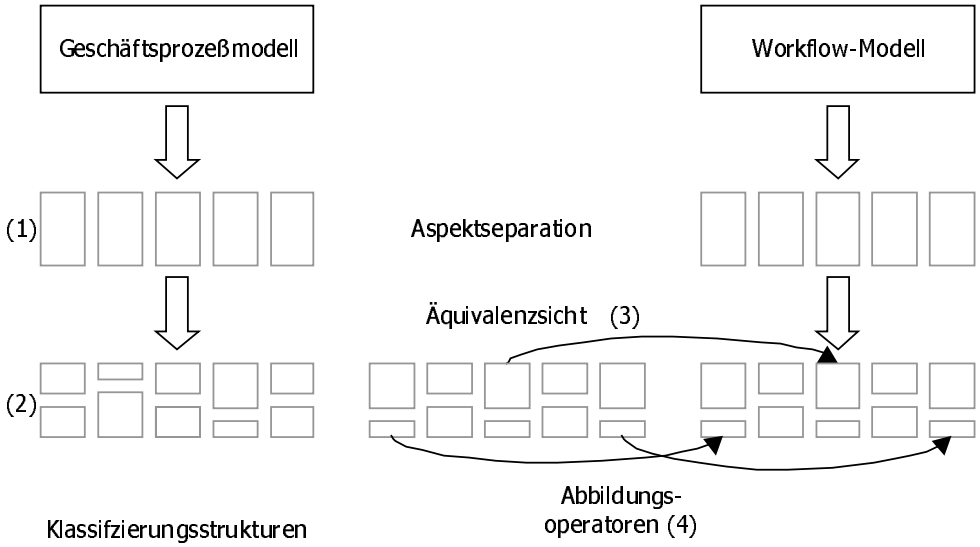


Abbildung 180: Inkrementelle Abbildung

13.4.3 Existierende Ansätze

Die existierenden Ansätze im Bereich der Abbildung von Geschäftsprozessen auf Workflow-Schemata sind allesamt gesamtabbildende Ansätze. Eine Änderung des Geschäftsprozeßschemas hat daher zur Folge, daß das gesamte Schema erneut abgebildet werden muß. Den existierenden Ansätzen ist es daher nicht möglich, die eigentlich von einer Änderung betroffenen Schemaelemente einzugrenzen und die erforderlichen erneuten Abbildungsoperationen auf diese zu begrenzen. Die tiefere Ursache für die Einschränkungen der existierenden Ansätze liegt in der zu groben und unvollständigen Gestaltung des Raster für das Geschäftsprozeß- und Workflow-Modell.

13.5 Zusammenfassung

In diesem Kapitel wurde nachgewiesen, daß der in dieser Arbeit entwickelte Ansatz zur Unterstützung weitreichender Prozesse in der Lage ist und damit einen Neuigkeitswert gegenüber bisherigen Ansätzen erzielt.

Dazu wurde die Erfüllung aller in Kapitel 3 identifizierten informationstechnischen Anforderungen nachgewiesen. Grundlage hierfür sind die Erfahrungen aus dem praktischen Ein-

satz der Konzepte in mehreren Projekten. Dabei konnte auch die Anwendbarkeit der Konzepte auf unterschiedliche Technologien nachgewiesen werden. So dienten im WOGE-, CompFlow- und VORREITER-Projekt unterschiedliche Technologien als Implementierungsgrundlage. Die Beiträge der einzelnen Lösungskonzepte zur Erfüllung der Anforderungen sind in Abbildung 181 dargestellt.

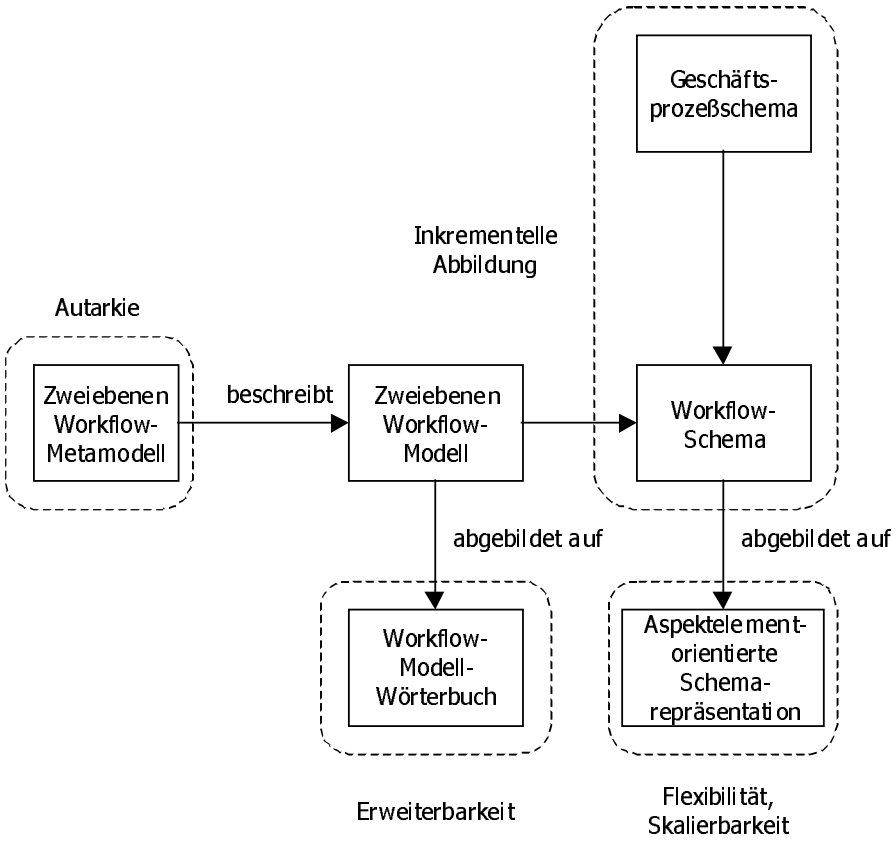


Abbildung 181: Anforderungserfüllung durch die entwickelten Lösungskonzepte

14 Zusammenfassung und Ausblick

Mit diesem Kapitel wird eine Zusammenfassung der wesentlichen Ergebnisse der Arbeit gegeben. Es beginnt mit einem kurzen Abriß der Ausgangssituation und der Rahmenbedingungen für die Unterstützung weitreichender Prozesse.

Danach wird schrittweise der Lösungsweg der Arbeit vorgestellt. Auf dieser Basis können die von dieser Arbeit erzielten Neuigkeitswerte dargestellt werden. An erster Stelle ist dabei das Konzept für die umfassende Unterstützung weitreichender Prozesse anzuführen. Darüber hinaus konnten weitere Fortschritte erzielt werden, wie beispielsweise die Schaffung der kompositen Anwendung.

Den Abschluß des Kapitels bildet ein Ausblick auf mögliche Erweiterungen des hier entwickelten Ansatzes zur Unterstützung ungeplanter Situationen und die Übertragung auf andere Bereiche wie elektronische Märkte.

14.1 Ausgangssituation

Die Zusammenarbeit von Unternehmen und Organisationen gewinnt zunehmend an volkswirtschaftlicher Bedeutung. Ihre einfachste Form ist die Ausführung von Geschäftsprozessen über mehrere Niederlassungen hinweg und reicht bis zu virtuellen Unternehmen. Durchgeführt wird diese Zusammenarbeit typischerweise in Form von weitreichenden (Geschäfts-)Prozessen. Bei der informationstechnischen Unterstützung von weitreichenden Geschäftsprozessen tauchen jedoch neue Rahmenbedingungen auf, die bei der Unterstützung lokal begrenzter Geschäftsprozesse noch nicht galten.

1. Bei weitreichenden Geschäftsprozessen herrscht die Autarkie der Prozeßteilnehmer. Sie sind nicht mehr, wie bei Geschäftsprozessen in einem Unternehmen, den Regeln einer Organisation unterworfen, sondern selbständig.

2. Bei der Unterstützung weitreichender Geschäftsprozesse ist mit häufigeren und tiefer greifenden Änderungen der Prozeßschemata zu rechnen als bisher. Dies erklärt sich aus der größeren Ausdehnung und größeren Teilnehmerzahl weitreichender Prozesse. Durch sie existieren auch deutlich mehr Änderungsquellen als bei der lokal begrenzten Ausführung von Geschäftsprozessen.

3. Die informationstechnische Unterstützung für weitreichende Geschäftsprozesse muß umfassend sein. D.h., es muß eine so leistungsfähige Prozeßunterstützung vorhanden sein, daß sich alle Prozesse eines Unternehmens oder einer Verwaltung un-

terstützen lassen. Sie muß ohne Abstriche bei anderen Anforderungen, insbesondere der flexiblen Umsetzung von geänderten Prozeßschemata, erfüllt werden.

14.2 Lösungsweg

Der verfolgte Lösungsweg ist schematisch in Abbildung 182 dargestellt. Aus den Rahmenbedingungen für die Unterstützung weitreichender Prozesse ergeben sich neue Anforderungen an die informationstechnische Prozeßunterstützung. Die organisatorische Autarkie der Prozeßteilnehmer erfordert es, die Autarkie auch bei der informationstechnischen Prozeßunterstützung zu berücksichtigen. Aus der Rahmenbedingung, daß Änderungen tiefer greifend sind als bisher, ergibt sich eine weitere Anforderung. Sie besagt, daß es für die informationstechnische Unterstützung weitreichender Prozesse erforderlich ist, nicht nur Anpassungen von Workflow-Schemata, sondern auch Erweiterungen des zu Grunde liegenden Workflow-Modells zur Laufzeit umsetzen können. Aus der Rahmenbedingung, daß Prozeßänderungen bei weitreichenden Prozessen häufiger auftauchen, ergeben sich zwei Anforderungen. Erstens muß für eine inkrementelle Abbildung der Geschäftsprozeß- auf Workflow-Schemata gesorgt werden, um eine schnelle Umsetzung von Änderungen zu gewährleisten. Zweitens müssen veränderte Workflow-Schemata schnell informationstechnisch umgesetzt werden, was sich in der Anforderung nach Flexibilität niederschlägt. Schließlich erfordert die Rahmenbedingung der umfassenden Prozeßunterstützung eine hohe Skalierbarkeit der informationstechnischen Prozeßunterstützung. Die Erfüllung einer dieser Anforderungen darf natürlich nicht in Widerspruch zu der Erfüllung der übrigen Anforderungen geraten. Zunächst werden daher die Anforderungen der autarken Zuweisung informationstechnischer Ressourcen, der Erweiterbarkeit des Workflow-Modells, sowie der Flexibilität und Skalierbarkeit verfolgt. Die Anforderung der inkrementellen Abbildung von Geschäftsprozeß- auf Workflow-Schemata wird unabhängig von den übrigen Anforderungen behandelt.

Bei der Analyse existierender Ansätze zeigte sich, daß keiner diese Anforderungen erfüllt. Zwar gibt es Ansätze, die einzelne Anforderungen erfüllen, doch versperren sie durch die von ihnen gewählten Lösungsansätze den Weg für die Erfüllung der übrigen Anforderungen. Ausgangspunkt für die Entwicklung des Lösungsansatzes war die Identifizierung von drei Problemgebieten, in denen die Schwächen der existierenden Ansätze begründet liegen. Die Problembereiche sind die Gestaltung des Workflow-Metamodells, die Repräsentation des Workflow-Modells und die Repräsentation von Workflow-Schemata. Für jeden dieser drei Problembereiche wurde ein Lösungskonzept entwickelt:

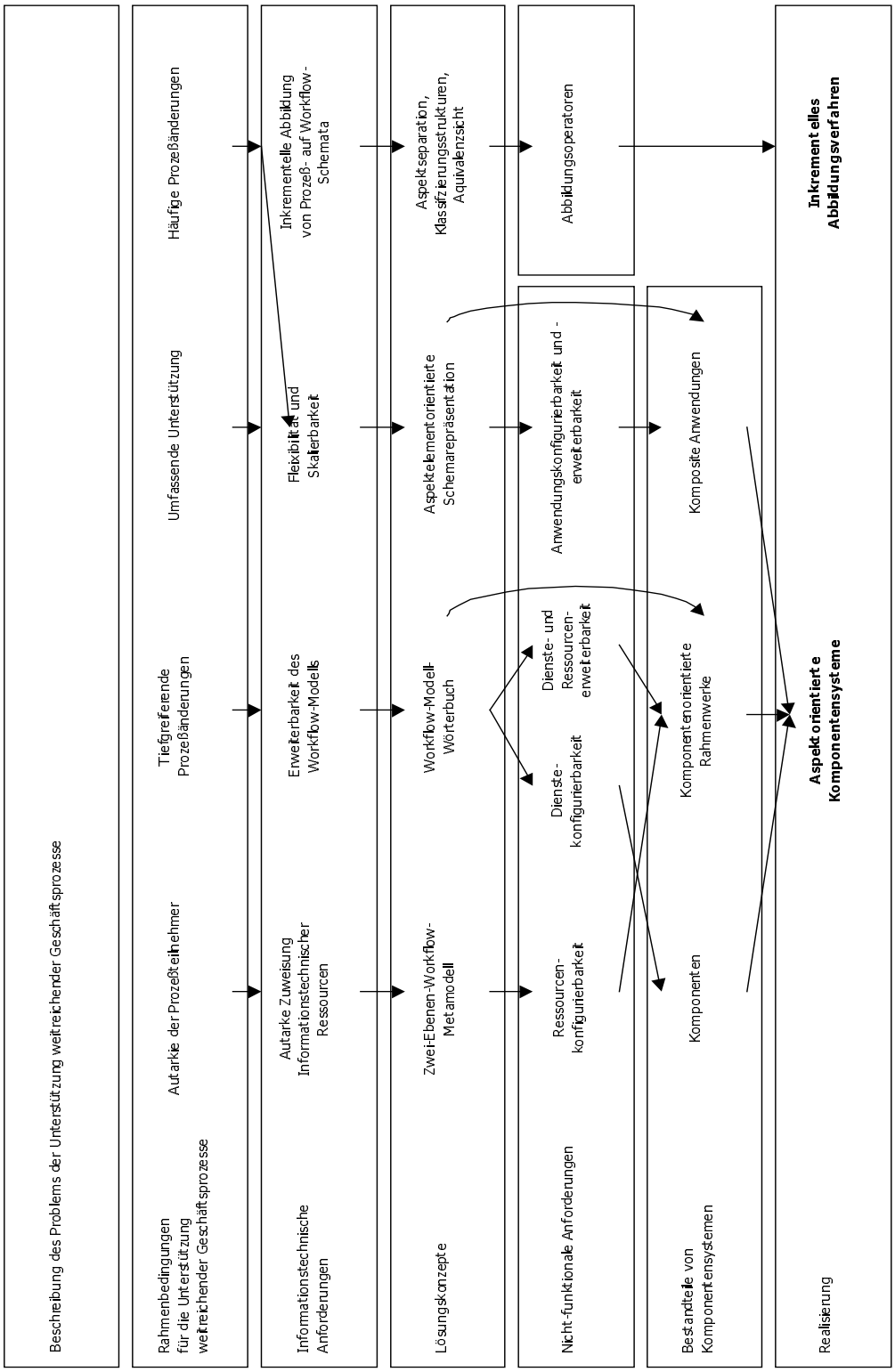


Abbildung 182: Herleitung der Lösungskonzepte

- Das Zwei-Ebenen-Workflow-Metamodell legt die Grundlage für die autarke und flexible Zuweisung informationstechnischer Ressourcen bei der Workflow-Unterstützung.
- Die dynamische Repräsentation des Workflow-Modells wird durch ein Workflow-Modell-Wörterbuch erreicht. Dieses ermöglicht die Erweiterung der unterstützen Workflow-Modellelemente zur Laufzeit.

- Die aspektelementorientierte Schemarepräsention legt die Grundlage für die flexible Umsetzung von Workflow-Schemaänderungen bei gleichzeitiger Skalierbarkeit. Darüber hinaus können auch Erweiterungen um Schemaelemente vorgenommen werden, die durch Nutzung des Workflow-Modell-Wörterbuchs neu aufgenommen worden sind.

Aus den Lösungskonzepten des Zwei-Ebenen-Workflow-Modells, des Workflow-Modell-Wörterbuchs und der aspektelementorientierten Schemarepräsentation ergeben sich nicht-funktionale Anforderungen. Diese müssen von einer geeigneten Software-Architektur erfüllt werden. So leitet sich aus dem Zwei-Ebenen-Workflow-Modell die Anforderung der Ressourcenkonfigurierbarkeit, aus dem Workflow-Modell-Wörterbuch die Anforderung der Dienstekonfigurierbarkeit sowie der Dienste- und Ressourcenerweiterbarkeit und aus der aspektelementorientierten Schemarepräsentation die Anwendungserweiterbarkeit und -konfigurierbarkeit ab.

Komponentensysteme erfüllen diese nicht-funktionalen Anforderungen. Sie bestehen aus Komponenten, komponentenorientierten Rahmenwerken und kompositen Anwendungen. Durch jeden dieser Bestandteile werden eine oder mehrere nicht-funktionale Anforderungen erfüllt. Komponenten erfüllen die Anforderung der Dienstekonfigurierbarkeit, komponentenorientierte Rahmenwerke erfüllen die Anforderungen der Ressourcenkonfigurierbarkeit sowie der Dienste- und Ressourcenerweiterbarkeit. Da die Anforderungen der Anwendungserweiterbarkeit und –konfigurierbarkeit weder von Komponenten noch von komponentenorientierten Rahmenwerken erfüllt wurden, wurde das Konzept der kompositen Anwendung entwickelt, das die Anforderungen erfüllt.

Auf der Basis der Komponentensysteme kann die Realisierung der Lösungskonzepte durchgeführt werden. So wird die aspektelementorientierte Schemarepräsentation durch komposite Anwendungen realisiert. Das Workflow-Modell-Wörterbuch kann Informationen aus der Registratur komponentenorientierter Rahmenwerke durch ein Integrationsmodell nutzen. Die sich ergebende Gesamtarchitektur wird als aspektorientiertes Komponentensystem bezeichnet.

Parallel dazu verlief die Schaffung einer inkrementellen Abbildung von Prozeß- auf Workflow-Schemata. Kernelemente sind die Aspektseparation der Metamodelle von Geschäftsprozeß- und Workflow-Modell, die Feinzerlegung der Modelle durch Klassifizierungsstrukturen und die Schaffung einer Äquivalenzsicht, die schließlich zur Ausbildung bijektiver Abbildungsoperatoren führt.

Die Implementierung im Rahmen mehrerer Projekte zeigte die Praxistauglichkeit der hier entwickelten Konzepte. Die beim operativen Einsatz der Konzepte gesammelten Erfahrungen waren die Grundlage für die Evaluierung die den Abschluß der Arbeit bildete.

14.3 Erzielter Neuigkeitswert

14.3.1 Informationstechnische Unterstützung weitreichender Prozesse

Der zentrale Neuigkeitswert dieser Arbeit liegt in der erstmaligen Möglichkeit zur Unterstützung weitreichender Prozesse. Im vorangegangenen Kapitel wurde nachgewiesen, daß der hier entwickelte Ansatz die obigen informationstechnischen Anforderungen aus der Unterstützung weitreichender Prozesse erfüllt. Kein existierender Ansatz erfüllt hingegen die gestellten Anforderungen in ihrer Gesamtheit, wie in Kapitel 4 gezeigt wurde.

Das Ziel wurde durch die Bereitstellung dreier Lösungskonzepte und eines inkrementellen Abbildungsverfahrens für Geschäftsprozesse erreicht. Bei den Lösungskonzepten handelt es sich um das Zwei-Ebenen-Workflow-Modell, das Workflow-Modell-Wörterbuch und die aspektele-

mentorientierte Schemarepräsentation. Der erzielte Neuigkeitswert wurde durch die praktische Umsetzung der Konzepte im Rahmen mehrerer Projekte nachgewiesen.

14.3.2 Weitergehende Beiträge

Beim Erreichen des Ziels der Unterstützung weitreichender Prozesse wurden eine Reihe von Beiträgen entwickelt, die auch für sich allein gesehen einen Neuigkeitswert besitzen. Auch hier wird der erzielte Fortschritt in Bezug zum bisherigen Stand der Technik dargestellt.

14.3.2.1 Schaffung eines begrifflichen Instrumentariums für die Beurteilung von prozeßunterstützenden Software-technischen Systemen

Bisher sind kaum Untersuchungen von Informationssystemarchitekturen zur Prozeßunterstützung durchgeführt worden. Erst recht fehlt ein begriffliches Instrumentarium für die Beurteilung von prozeßunterstützenden Software-technischen Systemen. Daher stellen auch die in dieser Arbeit eingeführten Begriffssysteme und Modelle einen Neuigkeitswert dar.

14.3.2.2 Komposite Anwendungen

In komponentenorientierten Rahmenwerken fehlte bisher eine erweiter- und konfigurierbare Anwendungsarchitektur. Diese Lücke wird durch die in dieser Arbeit entwickelte komposite Anwendung geschlossen. Sie erfüllt die Anforderung der unabhängigen Anwendungserweiterbarkeit und –konfigurierbarkeit. Für die Durchführung der Spezialisierung in kompositen Anwendungen wurden außerdem zwei neue Spezialisierungsmechanismen entwickelt, die anwendungsdokumentbasierte und die registraturbasierte Spezialisierung.

14.3.2.3 Terminologische Erfassung von Komponenten und komponentenorientierten Rahmenwerken

Es wurde ein neuer Begriffsapparat für Komponenten und komponentenorientierte Rahmenwerke entwickelt, der es insbesondere ermöglicht zu klären, welche der identifizierten Anforderungen von Komponenten und komponentenorientierten Rahmenwerken erfüllt werden.

14.4 Ausblick

14.4.1 Übertragbarkeit

Die hier vorgelegte Konzeption zur Unterstützung von Prozessen in Komponentensystemen kann Anregungen für die Gestaltung anderer Anwendungen wie beispielsweise elektronischer Märkte geben. So gibt es eine Reihe von ersten Ansätzen zur komponentenorientierten Gestaltung von elektronischen Märkten [Voge98], [ReBe99] [AFHL99], diesen fehlt jedoch eine zugrundeliegende Systematik. Die in dieser Arbeit entwickelten Konzepte eignen sich besonders für diese Aufgabe. So stellen elektronische Märkte die logische Ergänzung zur informationstechnischen Unterstützung von Geschäftsprozessen dar. Elektronische Märkte unterstützen die Anbahnungsphase von Geschäftsprozessen, indem sie ein elektronisches Forum für die Plazierung von Angeboten bereitstellen. Der Medienbruch, der sonst an der Schnittstelle zwischen Kunde und Anbieter auftritt, kann so vermieden werden.

Ein weiterer Bereich, in den die Ergebnisse dieser Arbeit übertragen werden können, ist der Übersetzer-Bau. So läßt sich die Übersetzung eines Programms als ein Workflow auffassen, bei dem das zu übersetzende Programm eine Reihe von Transformationen durchläuft. Durch Anwendung der Konzepte dieser Arbeit können Übersetzer aus flexibel kombinierbaren Komponenten zusammengesetzt werden. Die auf diese Weise entwickelten Übersetzer lassen sich deutlich einfacher als die bisherigen anpassen, um beispielsweise Optimierungen vorzunehmen.

14.4.2 Erweiterbarkeit

Die in der Arbeit entwickelten Konzepte haben sich auf die software-technische Ausgestaltung einer Architektur für die Unterstützung weitreichender Geschäftsprozesse konzentriert. Daher mußten Gesichtspunkte wie beispielsweise die Einbettung der entwickelten Verfahren in einen Software-Entwicklungsprozeß zurückstehen. Es bietet sich daher an, die in der Arbeit entwickelten Konzepte in einen Software-Entwicklungsprozeß einzubetten, wie er beispielsweise in INCOME [StOS93], [ScOS94], [OSSS94] beschrieben worden ist.

Wie bereits in Kapitel 2 kurz angesprochen, erfassen Geschäftsprozesse keinesfalls die gesamte betriebliche Realität, sondern bilden mit dem Ziel der Erfassung, Darstellung und Optimierung von Prozessen nur einen Ausschnitt ab. Geschäftsprozesse erfassen nur die Realitätsausschnitte für die planmäßige Durchführung von Prozessen. Bisher unbeachtet blieben außerhalb der Prozeßsicht liegende Realitätsbereiche wie ungeplante Situationen und Ereignisse. Diese können überaus einflußreich auf die Prozeßausführung sein, werden aber bisher in die Prozeßsicht nicht aufgenommen, da sie nur im Ausnahmefall für die Prozesse von Bedeutung werden. Hierfür ist die Modellierung des Umfelds von Geschäftsprozessen und die Schaffung zusätzlicher Aspekte und Aspektelemente zu deren informationstechnischer Unterstützung notwendig [Kuhn98]. Die in dieser Arbeit entwickelten Konzepte eignen sich in besonderer Weise für die Unterstützung von ungeplanten Situationen und Ereignissen, da die dafür notwendigen zusätzlichen Aspekte leicht integriert werden können. Basis hierfür ist das Workflow-Modell-Wörterbuch, das eine einfache Erweiterbarkeit um zusätzliche Aspekte wie beispielsweise solche für die Unterstützung ungeplanter Situationen und Ereignisse gewährleistet.

Index

F

G

H

I

J

K

L

M

U

V

W

Z

Literaturverzeichnis

[AAAK96] G. Alonso, D. Agrawal, A. El Abbadi, M. Kamath, R. Günthör, C. Mohan: Advanced Transaction Models in Workflow Contexts. ICDE, 1996, S. 574 — 581.

[AAAM97] G. Alonso, D. Agrawal, A. El Abbadi, C. Mohan: Functionality and Limitations of Current Workflow Management Systems. IEEE Expert (Special Issue on Cooperative Information Systems), 1997.

[AAEM95] G. Alonso, D. Agrawal, A. El Abbadi, C. Mohan, R. Günthör, M. Kamath: Exotica/FMQM: A persistent message-based architecture for distributed workflow-management. In Proceedings of IFIP WG 8.1 Workgroup Conference on Information Systems Development for Decentralized Organizations (ISDO95), Trondheim, Norway, August 1995.

[Adle95] R. M. Adler: Emerging Standards for Component Software. IEEE Computer, 28(3), März 1995, S. 68 — 73.

[AFHL99] G. Alonso, U. Fiedler, C. Hagen, A. Lazcano, H. Schuldt, N. Weiler: Wise: Business to Business E-Commerce. In Proceedings 9th International Workshop on Research Issues on Data Engineering (RIDE-VE'99), Sydney, Australia, 23. — 24. März, 1999.

[AGHA86] G. A., Agha: Actors. MIT Press, Cambridge, Massachusetts, 1996.

[AGMK95] G. Alonso, R. Günthör, Mohan U. Kamath, D. Agrawal, A. El Abbadi, C. Mohan: Exotica/FMDC: Handling disconnected clients in a workflow management system. In Proceedings of 3rd International Conference on Cooperative Information Systems (CoopIS-95), Vienna, Austria, Mai 1995.

[AkBe92] M. Aksit, L. Bergmans: Obstacles in Object-Oriented Software Development. In Proceedings OOPSLA 1992, S. 341 — 358.

[AkBV92] M. Aksit, L. Bergmans, S. Vural: An Object-Oriented Language-Database Integration Model: The Composition-Filters Approach. In Proceedings of the ECOOP '92 Conference, LNCS 615, Springer Verlag, Berlin, 1992, S. 372 — 395.

[AlSc96] G. Alonso, H. J. Schek: Research Issues in Large Workflow Management Systems. NSF Workshop on Workflow and Process Automation in Information Systems, Athens, Georgia, USA, Mai 1996.

[Ambe95] M. Amberg: Ableitung von Spezifikationen für Workflow-Managementsysteme aus Geschäftsprozeßmodellen, Informationssystem-Architekturen. Rundbrief des GI-Fachausschusses 5.2, 2(2), 1995.

[Ambe96a] M. Amberg: Transformation von Geschäftsprozeßmodellen des SOM-Ansatzes in workflow-orientierte Anwendungssysteme. In [BeRo96].

[Ambe96b] M. Amberg: Modelling Adaptive Workflows in Distributed Environments. In Proceedings First International Conference on Practical Aspects of Knowledge Management, Basel, Schweiz, 30. – 31. Oktober 1996.

[Ambe97] M. Amberg: The benefits of business process modelling for specifying workflow-oriented application-systems. In WfMC Handbook. John Wiley & Sons, Chicester, 1997, S. 61-68.

[AsSc97] U. Aßmann, R. Schmidt: Towards a Model For Composed Extensible Components. In Proceedings Workshop Foundations of Component-Based Systems, Zürich, Schweiz, 26. September, 1997, S. 23 — 30.

[Assm97] U. Assmann: Meta-programming Composers in Second Generation Component Systems, Technischer Bericht 17/97, Fakultät für Informatik der Universität Karlsruhe, 1997.

[ASSR93] P. Attie, M. Singh, A. Sheth, M. Rusinkiewicz: Specifying and Enforcing Intertask Dependencies. In Proceedings of the 19th Intl Conf. on Very Large Data Bases, August 1993.

[ASWe96] M. Amberg, R. Striemer, M. Weske: Modellierung von Workflows: Ein Rahmenmodell. Beitrag zum GI Arbeitskreis Workflow März 1996.

[AWBB93] M. Aksit, K. Wakita, J. Bosch, L. Bergmans, A. Yonezawa: Abstracting Object Interactions Using Composition Filters. In R. Guerraoui, O. Nierstrasz, M. Riveill (Eds.): Object-Based Distributed Programming, ECOOP '93 Workshop, Kaiserslautern, Germany, July 26-27, 1993. LNCS Vol. 791, Springer Verlag, Berlin, 1994.

[BaDa98] Th. Bauer, P. Dadam: Architekturen für skalierbare Workflow-Management-Systeme - Klassifikation und Analyse, Universität Ulm, Fakultät für Informatik, Ulmer Informatik-Berichte Nr. 98, 01, 1998.

[Balz96] H. Balzert: Lehrbuch der Software-Technik: Software-Entwicklung. Spektrum Verlag, Heidelberg 1996.

[Baue98] T. Bauer: Architekturen für skalierbare Workflow-Management-Systeme : Klassifikation und Analyse, Universität Ulm, Fakultät für Informatik, Ulmer Informatik-Berichte, Nr. 98, 02, 1998

[Baue99] T. Bauer: Veteilungsmodelle für Workflow-Management-Systeme : Klassifikation und Simulation, Universität Ulm, Fakultät für Informatik,Ulmer Informatik-Berichte, Nr. 99, 02, 1999

[BeDa94] P. A. Bernstein, U. Dayal: An Overview of Repository Technology. In Proceedings 20th Intl. Conf. on Very Large Data Bases, Santiago, Chile 1994, S. 705 — 713.

[Bern96] P. A. Bernstein: Repository Systems Engineering, Tutorial. In H. V. Jagadish, I. S. Mumick (Eds.): Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data, Montreal, Quebec, Kanada, June 4-6, 1996. ACM Press 1996, SIGMOD Record, 25(2), Juni 1996.

[Bern98] P. A. Bernstein; Repositories and Object-Oriented Databases, SIGMOD Record 98, März 1998, S. 34 — 46.

[BeRo96] J. Becker, M. Rosemann (Hrsg.): Workflow-Management - State-of-the-Art aus Sicht von Theorie und Praxis. Arbeitsbericht 47, Institut für Wirtschaftsinformatik, Universität Münster, 1996.

[BeSR95] J. Becker, R. Schütte, M. Rosemann: Grundsätze ordnungsmäßiger Modellierung. Wirtschaftsinformatik, 37(5), 1995.

[BeVo96] J. Becker, G. Vossen: Geschäftsprozeßmodellierung. Dpunkt Verlag, Heidelberg, 1996.

[BHJS96] C. Bußler, P. Heinl, S. Jablonski, H. Schuster, K. Stein: Das WWW als Benutzerschnittstelle und Basisdienst zur Applikationsintegration für Workflow-Management-Systeme. In: Jeusfeld, M. (Hrsg.): Informationsserver für das Internet. Anforderungen, Konzepte, Methoden. In Proceedings EMISA-Fachgruppentreffen, Aachen, Oktober, 1996.

[BHSS97] P. A. Bernstein, B. Harry, P. J. Sanders, D. Shutt, J. Zander: The Microsoft Repository, Invited Keynote address. In Proceedings of the 23th Intl. Conf. on Very Large Data Bases, Athen, Griechenland, 1997, S. 3 — 12.

[BJPW99] A. Beugnard, J. Jézéquel, N. Plouzeau, D. Watkins: Making Components Contract Aware. IEEE Computer, 32(7), Juli 1999, S. 38 — 45

[Böhm97] M. Böhm. Objektorientierte Implementierungstechniken für Workflow-Management-Systeme in OMA-konformen Architekturen. In [JaBS97].

[BöMS97] M. Böhm, K. Meyer-Wegener, W. Schulze: Unterstützung der Workflow-Entwicklung durch ein unternehmensweites Repository für Geschäftsprozeßrealisierungen. In [Kral97].

[Box98] D. Box: Essential COM. Addison-Wesley, Reading, Massachusetts, 1998.

[Broc96] K. Brockschmidt: Inside OLE 2nd Edition. MS Press, Redmond, 1996.

[BuJa95] C. Bussler, S. Jablonski. Scalability and extensibility through modularity: Architecture of the Mobile workflow management system. In Proceedings 5th Workshop on Information Technologies and Systems (WITS '95), Amsterdam, Niederlande, 9. –10. Dezember, 1995.

[BuJa96] C. Bußler, S. Jablonski: Die Architektur des modularen Workflow-Management-Systems Mobile. In [VoBe96].

[CCPP96a] F. Casati, S. Ceri, B. Pernici, G. Pozzi: Workflow Evolution. In Proceedings of the 15th ER96 Int. Conf., Cottbus, 1996

[CCPP96b] F. Casati, S. Ceri, B. Pernici., G.Pozzi: Conceptual WorkFlow Modeling. In Proceedings Object-Oriented and Entity-Relationship Int. Conf. , Goldcoast, Australia, 1995, S. 341 — 354.

[CCPP98] F. Casati, S. Ceri, B. Pernici, G. Pozzi. Workflow Evolution, Data & Knowledge Engineering, 24(3), Januar 1998.

[CeGS97] S. Ceri, P. Grefen, G. Sánchez: WIDE - A Distributed Architecture for Workflow Management. RIDE97. Seventh International Workshop on Research Issues in Data Engineering, Birmingham, England, April 1997.

[CGPP97] F. Casati, P. Grefen, B. Pernici, G. Pozzi, G. Sánchez. WIDE Workflow model and architecture, http://dis.sema.es/projects/WIDE/Documents/ase30_4.ps.gz.

[Chap96] D. Chappell: Understanding ActiveX and OLE. Microsoft Press, Redmond, 1996.

[ChLK99] D. K. W. Chiu, Q. Li, K. Karlapalem: A Meta Modelling Approach To Workflow Management Systems Supporting Exception Handling. Information Systems, 24(2), 1999, S. 159 — 184.

[ChRa90] P. K. Chrysanthis, K. Ramamritham: ACTA: A Framework for Specifying and Reasoning about Transaction Structure and Behaviour. In Proceedings ACM SIGMOD International Conference on the Management of Data, Atlantic City, NJ, Mai 1990, S. 194 — 203.

[CiSc96] O. Ciupke, R. Schmidt: Components As Context-Independent Units of Software. WCOP 96, Linz 1996. In Special Issues in Object-Oriented Programming, Workshop Reader of the 10th European Conference on Object-Oriented Programming ECOOP96, Dpunkt Verlag, Heidelberg, 1996, S. 139 – 143.

[COM] http://www.microsoft.com/com/

[Cox90] B. J. Cox: Planning the software industrial revolution. IEEE Software, 7(6), 1990.

[CROS] http://www.crossflow.org/

[CuKO92] B. Curtis, M. I. Kellner, J. Over: Process Modelling: Communications of the ACM, September 1992, 35(9), S. 75 — 90.

[CVSG96] D. Chan, J. Vonk, G. Sánchez, P. Grefen, P. Apers: A Specification Language for the WIDE Workflow Model. In Proceedings CaiSE97, Barcelona, 1997.

[DaRe98] P. Dadam, M. Reichert: The ADEPT WfMS Project at the University of Ulm, Germany. In Proceedings 1st European Workshop on Workflow and Process Management (WPM'98). "Workflow Management Research Projects" 2. Oktober, 1998, Swiss Federal Institute of Technology (ETH) Zürich, Schweiz.

[DASB98] A. Dogac, G. Alonso, H.-J. Schek, C. Beeri: The MARIFLOW Project: A Workflow Management System for the Maritime Industry. In Proceedings 1st European Workshop on Workflow and Process Management (WPM'98). "Workflow Management Research Projects" October, 2nd, 1998 Swiss Federal Institute of Technology (ETH) Zürich, Schweiz.

[Dave93] T. Davenport: Process Innovation - Reengineering Work through Information Technology. Harvard Business School Press, Boston/Massachusetts, 1993.

[Davi99] M. David: Transaktionale Workflows auf der Basis von Enterprise JavaBeans. Diplomarbeit an der Universität Karlsruhe, 1999

[DeGS95] W. Deiter, V. Gruhn, R. Striemer: Der FUNSOFT-Ansatz zum integrierten Geschäftsprozeßmanagement. Wirtschaftsinformatik 37(5), 1995, S. 459 — 466.

[Deri96] L. Deri: Droplets: Breaking Monolithic Applications Apart. In Proceedings Workshop on Workshop on Composability Issues in Object-Orientation, Linz, 1996.

[DeVÖ96] M. Derungs, P. Vogler, H. Österle: Metamodell Workflow. Report Nr. HSG/CC PSI/3. Version 1.5. St. Gallen 1996 (http://www-iwi.unisg.ch/iwi2/cc/psi/abpsi3.html).

[DKMS96] S. Das, K. Kochut, J. Miller, A. Sheth, D. Worah: ORBWork: A Reliable Distributed CORBA-based Workflow Enactment System for METEOR2. Technical report, University of Georgia, Athens 1996.

[DoZR97] E. Dominguez, M. A. Zapata, J. Rubio: A Conceptual Approach to Meta-Modelling. In A. Olivé, J. A. Pastor (Hrsg.): Advanced Information Systems Engineering. 9th International Conference, CaiSE'97, Berlin, Springer-Verlag, 1997.

[Dros89] G. Drosdowski (Hrsg.): Duden – Das Herkunftswörterbuch der deutschen Sprache. Dudenverlag Mannheim, 1989.

[EdGL96] J. Eder, H. Groiss, W. Liebhart: Workflow-Systeme im WWW, ADV Kongreß, Wien, 1996.

[EJB] http://java.sun.com/ejb

[Elma92] A. Elmagarmid (ed.): Database Transaction Models for Advanced Applications. Morgan Kaufmann Publ., San Mateo, CA, 1992

[EnWa98] A. Engelhardt, C. Wargitsch: Scaling Workflow-Applications with Component and Internet Technology: Organizational and Architectural Concepts. In Proceedings of the 31st Hawaii International Conference on System Sciences, Vol. IV, Los Alamitos 1998, S. 374 — 383.

[Fers94] A. Ferscha: Qualitative and quantitative analysis of business workflows using generalized stochastic petri nets. In G. Chroust, A. Benczur, editors. In Proceedings of CON '94, Workflow Management: Challenges, Paradigms and Products, Linz, Austria, R. Oldenbourg Verlag, München, 1994, S. 222 — 234.

[FeSi93] O. K. Ferstl, E. J. Sinz: Geschäftsprozeßmodellierung. Wirtschaftsinformatik, 35(6), 1993, S. 589 — 592.

[FeSi94] O. K. Ferstl, E. J. Sinz: From Business Process Modelling to the Specification of Distributed Business Application Systems - An Object-Oriented Approach. Bamberger Beiträge zur Wirtschaftsinformatik Nr. 20, Bamberg, 1994.

[FeSi96a] O. K. Ferstl, E. J. Sinz: Flexible Organizations Through Object-oriented and Transaction-oriented Information Systems. Bamberger Beiträge zur Wirtschaftsinformatik Nr. 37, Bamberg, 1996.

[FeSi96b] O. K. Ferstl, E. J. Sinz: Geschäftsprozeßmodellierung im Rahmen des Semantischen Objektmodells. In [BeVo96].

[Flow] http://www.ibm.com/flowmark

[FSHS97] O. K. Ferstl, E. J. Sinz, Ch. Hammel, M. Schlitt, S. Wolf: Bausteine für komponenten-basierte Anwendungssysteme. In HMD - Theorie und Praxis der Wirtschaftsinformatik. Heft 197, 1997, S. 24 – 46.

[GACT97] E. Gokkoca, M. Altinel, I. Cingil, E. N. Tatbul, P. Koksal, A. Dogac: Design and Implementation of a Distributed Workflow Enactment Service. In Proceedings of Intl. Conf on Cooperative Information Systems, Charleston, USA, Juni 1997.

[Gail95] G. E. Kaiser: Cooperative Transactions for Multiuser Environment. In W. Kim (Hrsg.): Modern Database Systems, ACM Press, New York 1995.

[GaSh93] D. Garlan, M. Shaw. An Introduction to Software Architecture: Advances in Software Engineering and Knowledge Engineering Vol. I. World Scientific Publishing, 1993.

[GaSh94] D. Garland, M. Shaw: An Introduction to Software Architecture. Technical Report CMU/SEI-94-TR-21, Carnegie Mellon University 1994.

[GeHo94] D. Georgakopoulos , M. F. Hornick. A framework for enforceable specifications of extended transaction models and transactional workflows. International Journal of Cooperative Information Systems, September 1994.

[GeHS95] D. Georgakopoulos, M. Hornick, A. Sheth: An Overview of Workflow Management: From Process Modelling to Workflow Automation Infrastructure, Distributed and Parallel Databases, Kluwer Academic Publishers, September 1995.

[GeKT97] A. Geppert, M. Kradolfer, D. Tombros: Market-Based Workflow Management. Technical Report 97.15, Department of Computer Science, University of Zürich, 1997.

[Geor94] D. Georgakopoulos: Workflow Management: Concepts, Commercial Products and Infrastructure for Supporting Reliable Workflow Application Processing. Technical Report, GTE Corporation, TR-0284-12-94-165, 1994.

[GeTD98] A. Geppert, D. Tombros, K. R. Dittrich: Defining the Semantics of Reactive Components in Event-Driven Workflow Execution with Event Histories. Information Systems, 23(4), 1998, S. 235 — 252.

[GeTo97] A. Geppert, D. Tombros: Ereignisgesteuerte Workflow-Ausführung in verteilten Umgebungen. In [JaBS97].

[GeTo98] A. Geppert, D. Tombros: Event-based Distributed Workflow Execution with EVE. Technical Report 96.05 (revised 1998). Department of Computer Science, University of Zürich, 1998.

[Goos99] G. Goos: Vorlesungen über Informatik, Band 2, Objektorientiertes Programmieren und Algorithmen. Springer Verlag, Berlin, 1999.

[GrKa96] V. Gruhn und M. Kampmann, Modellierung unternehmensübergreifender Geschäftsprozesse mit FUNSOFT-Netzen. Wirtschaftsinformatik, 38(4), 1996, S. 383 — 390.

[HaCh93] M. Hammer, J. Champy: Reengineering the Cooperation, A Manifesto for Business Revolution. Harper Business, New York, 1993.

[HaCh95] M. Hammer, J. Champy: Business Reengineering : Die Radikalkur für das Unternehmen, 5. Aufl, Campus Verlag, Frankfurt, 1995.

[Hare88] D. Harel: On Visual Formalisms. Communications of the ACM, 31(5), Mai 1988, S. 514 – 530.

[HaSy95] U. Hasenkamp, M. Syring: CSCW – Computer Supported Cooperative Work in Organisationen – Grundlagen und Probleme in [HaKS95].

[HePe97] G. Herrmann, G. Pernul: Zur Bedeutung von Sicherheit in interorganisationellen Workflows. Wirtschaftsinformatik, 39(3), 1997, S. 217 — 224.

[HeRe99] D. Heuzeroth, R. Reussner: Dynamic Coupling of Binary Components and its Technical Support to be presented at GCSE'99 Young Researchers Workshop.

[Heri98] M. Hering: Abbildung von SOM-Geschäftsprozeßmodellen auf komponentenbasierte Workflow-Management-Systeme. Diplomarbeit an der Universität Karlsruhe, 1998.

[HoSW96] R. Holten, R. Striemer, M. Weske: Darstellung und Vergleich von Vorgehensmodellen zur Entwicklung von Workflow-Anwendungen. ISST Bericht 34/96, Berlin: Fraunhofer ISST, 1996.

[Jabl94] S. Jablonski. Mobile: A modular workflow model and architecture. In Proceedings of the Fourth International Working Conference on Dynamic Modelling and Information Systems, Noordwijkerhout, The Netherlands, September 1994.

[Jabl95a] S. Jablonski: Workflow-Management-Systeme – Modellierung und Architektur. International Thomson Computer Press, Bonn, 1995.

[Jabl95b] S. Jablonski: Workflow-Management-Systeme: Motivation, Modellierung, Architektur. Informatik Spektrum, 1995(18), S. 13 — 24.

[Jabl95c] S. Jablonski: On the Complementarity of Workflow Management and Business Process Modelling. SIGOIS Bulletin, 16(1), August 1995, S. 33 — 38.

[JaBS97] S. Jablonski, M. Böhm, W. Schulze: Workflow-Management: Entwicklung von Systemen und Anwendungen — Facetten einer neuen Technologie. Dpunkt Verlag, Heidelberg, 1997.

[JaBu96] S. Jablonski, C. Bussler: Workflow Management Modeling Concepts, Architecture and Implementation. International Thomson Computer Press. London 1996.

[Jaza95] M. Jazayeri: Component Programming – a fresh look at software components. Technischer Bericht der Universität Wien, TUV-1841-95-01, 1995.

[JNDI] http://java.sun.com/products/jndi

[Joha91] R. Johansen: Leading Business Teams. Addison-Wesley, Reading, Massachusetts, 1991.

[JTS] http://java.sun.com/products/jts

[KAGM96] M. Kamath, G. Alonso, R. Günthör, C. Mohan: Providing high availability in very large workflow management systems. In P. Apers, M. Bouzeghoub, G. Gardarin (Eds.): Proceedings 5th Int. Conference on Extending Database Technology (EDBT96), Avignon, France, Springer, 1996.

[KaRa95] M. U. Kamath, K. Ramamritham: Modelling, correctness & systems issues in supporting advanced database applications using workflow management systems. Technical report, University of Massachusetts, Juni 1995.

[KaRa96] M. U. Kamath, K. Ramamritham: Bridging the gap between transaction management and workflow management. NSF Workshop on Workflow and Process Automation in Information Systems, Athens, Georgia, Mai 1996.

[Kicz96] G. Kiczales: Aspect-oriented programming. ACM Computing Surveys, 28(4), Dezember 1996.

[KILL97] G. Kiczales, J. Irwin, J. Lamping, J.M. Loingtier, C. V. Lopes, C. Maeda, a. Mendhekar: Aspect-Oriented Programming. In Proceedings of the European Conference on Object-Oriented Programming (ECOOP), Finland. LNCS 1241, Springer Verlag, Berlin, 1997.

[KLSS94] T. Kirsche, R. Lenz, R. Schmidt, H. Schuster. Ereignisorientierte Spezifikation und Ausführung von (transaktionalen) Abläufen. In Proceedings 5. Workshop Transaktionskonzepte, Schloß Dhaun, Deutschland, Januar 1994.

[Kraj] M. Krajnc: What is component-oriented programming. http://www.odateam.com/cop/copwhats.html

[Kral97] H. Krallmann (Hrsg.): Wirtschaftsinformatik 97, Internationale Geschäftstätigkeit auf der Basis flexibler Organisationsstrukturen und leistungsfähiger Informationssysteme. Physica-Verlag, Heidelberg, 1997.

[Kreu96] S. Kreuser: Ableitung von Workflow-Spezifikationen aus dem Geschäftsprozeß-modell der Erstauftragsabwicklung eines Maschinenbau-Unternehmens. Diplomarbeit an der Universität Bamberg, 1996

[Kuhn98] E. Kuhn: Robuste Workflow-Managementsysteme. Interner Workshop am IPD der Universität Karlsruhe. 1998.

[LaLo95] S. M. Lang, P. C. Lockemann: Datenbankeinsatz. Springer Verlag, Berlin, 1995

[Lang96] G. Lang: Konzeption der Abbildung von Geschäftsprozeßmodellen nach dem SOM-Ansatz in Spezifikationen des Workflowmanagment-Systems WorkParty. Diplomarbeit an der Universität Bamberg, 1996.

[Lehm99] F. Lehmann: Repositorium-unterstützte Entwicklung von Workflow-Management-Anwendungen. Fachgebiet Wirtschaftsinformatik I der Technischen Hochschule Darmstadt, Arbeitsbericht 99, 01, 1999

[LeRo97] F. Lehmann, D. Roller: Workflow-based applications. IBM Systems Journal, 36(1), 1997.

[Lieb96] K. J. Lieberherr. Adaptive Object-Oriented Software: The Demeter Method with Propagation Patterns. PWS-Kent Publishing, 1996.

[LKSS96] P. Loos, O. Krier, P. Schimmel, A.-W. Scheer: WWW-gestützte überbetriebliche Logistik - Konzeption des Prototyps WODAN zur Kopplung von Beschaffungs- und Vertriebssystemen. Veröffentlichungen des Instituts für Wirtschaftsinformatik, Universität Saarbrücken, Heft 126, April 1996.

[Loos96] P. Loos: Workflow und industrielle Produktionsprozesse – Ansätze zur Integration. Veröffentlichungen des Instituts für Wirtschaftsinformatik, Universität Saarbrücken, Heft 123, Januar 1996.

[LoWa95] P. C. Lockemann, H. D. Walter: Object-Oriented Protocol Hierarchies for Distributed Workflow Systems. In [PaTo95].

[LuHo99] H. Ludwig, Y. Hoffner: Contract-based Cross-Organisational Workflows - The CrossFlow Project. Workshop on Cross-Organisational Workflow Management and Co-ordination, San Francisco, Februar 1999.

[Luko96] T. Lukosch: Konzeption der Abbildung von Geschäftsprozeßmodellen nach dem SOM-Ansatz in Spezifikationen des Workflow-Managementsystems FlowMark. Diplomarbeit an der Universität Bamberg, 1996.

[MAAA95a] C. Mohan, D. Agrawal, G. Alonso, A. El Abbadi, R. Günthör, M. Kamath. Exotica: A Project on Advanced Transaction Management and Workflow Systems. SIGOIS Bulletin, August 1995, 16(1), S. 45 — 50.

[MaCr94] T. W. Malone, K. Crowston: The Interdiciplinary Study of Coordination. ACM Computing Surveys, 26(1), März 1994, S. 87 — 119.

[MAGK95b] C. Mohan, G. Alonso, R. Günthör, Mohan U. Kamath: Exotica: A research perspective on workflow management systems. Bulletin of the Technical Comittee on Data Engineering, 18(1), März 1995, S. 19 — 26.

[MAGK95c] C. Mohan, G. Alonso, R. Günthör, M. U. Kamath, B. Reinwald: An Overview of the Exotica Research Project on Workflow Management Systems. In Proceedings 6th Int'l Workshop on High Performance Transaction Systems, Asilomar, September 1995.

[MARI] http://www.srdc.metu.edu.tr/mariflow/

[McIl69] M. D. McIlroy: Mass Produced Software Components. In Software Engineering, ed. P. Naur, B. Randell: NATO Science Committee, Januar 1969, S. 138 — 150.

[MCPM98] G. R. Malan, J. Chaar, S. Paul, P. Masters: eFlow: A JAVA-BASED WORKFLOW SERVICE. In Proceedings of the OOPSLA '98 Workshop on Implementation and Application of OO Workflow Management Systems, October 1998.

[Medv97a] N. Medvidovic. A Classification and Comparision Framework for Software Architecture Despcription Languages. Technical Report, UCI-ICS-97-02, University of Californica, Irvine, Januar 1997.

[Medv97b] N. Medvidovic: ADLs and Dynamic Architecture Changes. In Proceedings of the Second International Software Architecture Workshop (ISAW-2), San Francisco, Oktober 14-15, 1996, S. 24-27.

[MeFa95] P. Mertens, W. Faisst: Virtuelle Unternehmen, eine Organisationsstruktur für die Zukunft ? Technologie & Management, 44(2), 1995, S. 61 — 68.

[MeMi99] B. Meyer, C. Mingins: Component-Based Development: From Buzz to Spark. IEEE Computer, 32(7), Juli 1999, S. 38 — 45

[Mert94] P. Mertens: Virtuelle Unternehmen, Wirtschaftsinformatik, 36(2), 1994, S. 169 — 172.

[Meye99] B. Meyer: On To Components. IEEE Computer, 32(1), Januar 1999, S. 139 — 140.

[MiSc98] Ch. Mittasch, A. Schill: A Generic Workflow Environment based on CORBA Business Objects. Middleware'98, The Lake District, England, September 1998. In: Davis, N.; Raymond, K.; Seitz, J.: Middleware, IFIP Int. Conf. on Distributed System Platforms and Open Distributed Processing. London: Springer 1998, S. 19 — 34.

[MiSe98] L. Mikhajlov, E. Sekerinski: A Study of The Fragile Base Class Problem. ECOOP’98, 12th European Conference, Brüssel, Juli 1998. LNCS 1445, Springer Verlag, Berlin, S. 355 — 382.

[MKMG97] R. T. Monroe, A. Kompanek, R. Melton, D. Garlan: Architectural Styles, Design Patterns and Objects. IEEE Software, 14(1), Januar 1997, S. 43 — 52.

[Moha96] C. Mohan: Tutorial: State of the Art in Workflow Management System Research and Products. 5th International Conference on Extending Database Technology, Avignon, France, März 1996.

[MPSK98] J. A. Miller, D. Palaniswami, A. P. Sheth, K. Kochut, H. Singh: WebWork: METEOR2's Web-Based Workflow Management System. Journal of Intelligent Information Systems (JIIS), 10(2), 1994, S. 185 — 215.

[MSKW96] J. A. Miller, A. P. Sheth, K. J. Kochut, X. Wang: Corba-based run-time architectures for workflow management systems. Journal of Database Management, Special Issue on Multidatabases, Juli 1996.

[MWGW98] P. Muth, J. Weissenfels, M. Gillmann, G. Weikum: Mentor-lite: Integrating Light-Weight Workflow Management Systems within Business Environments. Integrating Light-Weight Workflow Management Systems within Existing Business Environments. ICDE 1999, S. 286 — 293.

[NiDa95] O. Nierstrasz, L. Dami: Component-Oriented Software Technology in [NiTs95].

[Nier91] O. Nierstrasz: The next 700 concurrent object-oriented languages – reflections on the future of object-based concurrency. In Tsichritzits D. (ed.) Object Composition. Centre Universitaire d'Informatique, Technischer Bericht, Universität Genf.

[NiLu97] O. Nierstrasz, M. Lumpe: Komponenten, Komponentenframeworks und Gluing. In HMD - Theorie und Praxis der Wirtschaftsinformatik, Heft 197/1997, S. 24 – 46.

[NiTs95] O. Nierstrasz, D. Tsichritzis (ed.). Object-orientied software composition, Prentice Hall, Hemel Hempstead 1995 .

[Note] http://www.lotus.com/home.nsf/welcome/notes

[Ober95] A. Oberweis: Modellierung und Ausführung von Workflows mit Petri-Netzen. Teubner Verlagsgesellschaft, Stuttgart, 1996.

[Ober96a] A. Oberweis: Verteilte betriebliche Abläufe und komplexe Objektstrukturen. Ein integriertes Modellierungskonzept für Workflow-Management-Systeme. Teubner Verlagsgesellschaft, Stuttgart, 1996.

[Ober96b] A. Oberweis: An integrated approach for the specification of processes and related complex structured objects in business applications. Decision Support Systems, Vol. 17, 1996, S. 31 — 53.

[Ober96c] http://www.oberon.ch/docu/whitepaper.html

[ObSS94] A. Oberweis, G. Scherrer, W. Stucky: INCOME/STAR: Methodology and Tools for the Development of Distributed Information Systems. Information Systems, 19(8), 1994, S. 643 — 660.

[ObSZ96] A. Oberweis, W. Stucky, G. Zimmermann: INCOME/STAR: Facing the challenges for cooperative information system development environments, in: W. König, K. Kurbel, P. Mertens, D. Pressmar (Hrsg.): Distributed Information Systems in Business, Springer Verlag, Berlin, 1996, S. 17 — 34.

[ObWS94] A. Oberweis, T. Wendel, W. Stucky: Teamwork Coordination in a Distributed Software Development Environment. In Proceedings of the IFIP '94 Workshop FG9: Communication and Coordination in Distributed Corporate Application Systems. Hamburg, 28. August – 2. September, 1994.

[OMG] http://www.omg.org

[OMGM] http://www.dstc.edu.au/Meta-Object-Facility/

[OMGW] http://wwwdb.inf.tu-dresden.de/wf-wg/

[OMTR98] P. Oriezy, N. Medvidovic, R. N. Taylor, D. S . Rosenblum: Software Architecture and Component Technologies: Bridging the gap. In Proceedings Workshop on Compositional Software Architectures, Monterey, Kalifornien 1998.

[OSSS94] A. Oberweis, V. Sänger, G. Scherrer, W. Stucky, T. Wendel, W. Weitz: Income/Star: Entwicklungs- und Wartungsumgebung für verteilte betriebliche Informationssysteme. Technischer Bericht, Institut für Angewandte Informatik und formale Beschreibungsverfahren, Universität Karlsruhe, Mai 1994.

[OSSW97] A. Oberweis, R. Schätzle, W. Stucky, W. Weitz, G. Zimmermann: INCOME/WF - A Petri Net Based Approach to Workflow Management, in: H. Krallmann (Hrsg): Wirtschaftsinformatik '97, Springer-Verlag, 1997, S. 557 — 580.

[PaPC97] S. Paul, E. Park, J. Chaar: Essential Requirements for a Workflow Standard. OOPSLA 97, Workshop on Business Object Design and Implementation.

[PaTo95] R. Pareschi, M. Tokoro: TAPOS Theory And Practice of Object Systems, 1(4), 1995.

[PfSz96] C. Pfister C. Szyperski: Why Objects Are Not Enough Proceedings, First International Component Users Conference (CUC'96), München, 15. – 19. Juli 1996.

[Pyth] http://www.python.org/

[ReBe99] B. Reich, I. Ben-Shaul. A Componentized Architecture for Scalable Market Exchange. International Workshop on Component-based Electronic Commerce, Berkely, Kalifornien, 25. Juli, 1998.

[ReDa98] M. Reichert, P. Dadam: ADEPTflex - Supporting Dynamic Changes of Workflows Without Loosing Control. Journal of Intelligent Information Systems (JIIS), Special Issue on Workflow Management Systems, 10(2), 1998, S. 93 — 129.

[Rein93] B. Reinwald: Workflow-Management in verteilten Systemen. Teubner Verlagsgesellschaft, Stuttgart, 1993.

[Rein95] B. Reinwald: Workflow Management, Tutorial. Deutsche Informatik Akademie, München, Dezember 1995.

[Riem98] G. Riempp: Wide Area Workflow Management. Springer Verlag, London, 1998.

[RiNa97] G. Riempp, L. Nastansky: From islands to flexible business process networks - enabling the interaction of distributed workflow management systems, in: Galliers, R. et al. (Eds.): Proceedings of the 5th European Conference on Information Systems, Cork, Vol. I, 1997, S. 481 — 496.

[RMI] http://java.sun.com/products/jdk/rmi/index.html

[RoMü96] M. Rosemann, M. zur Mühlen: Der Lösungsbeitrag von Metadatenmodellen beim Vergleich von Workflowmanagementsystemen. Arbeitsbericht Nr. 48 des Instituts für Wirtschaftsinformatik der Westfälischen Wilhelms-Universität Münster. Juni 1996.

[Rüpi97] A. Rüping. Software-Entwicklung mit objektorientierten Frameworks. Shaker Verlag, Aachen, 1997.

[RuSh94] M. Rusinkiewicz, A. Sheth: Specification and Execution of Transactional Workflows. In: Won Kim (Ed.): Modern Database Systems: The Object Model, Interoperability, and Beyond. ACM Press and Addison-Wesley, 1994, S. 592 — 620.

[SABK98] R. Schmidt, U. Assmann, P. Biegler, R. Kramer, P. C. Lockemann, C. Rolker: The Interrelatedness of Component-oriented Systems And Workflow-Management: What they can do for each other. Workshop on Compositional Software Architectures. Monterey, USA, 1998, http://www.objs.com/workshops/ws9801/papers/paper099.doc

[SBMW96] W. Schulze, M. Böhm, K. Meyer-Wegener: Services of Workflow Objects and Workflow Meta-Objects in OMG-compliant Environments, OOPSLA 96, Workshop on Business Object Design and Implementation, San José, CA.

[ScAs98a] R. Schmidt, U. Assmann: Component- and connector-based workflow systems. In Proceedings SAC98, Atlanta, USA, 1998.

[ScAs98b] R. Schmidt, U. Assmann: CompFlow: A Component-based and Aspect-separated Architecture for Workflow Support. In Proceedings EDBT ´98 Workshop on Workflow Management Systems, Valencia, 1998.

[ScAs98c] R. Schmidt, U. Assmann. Extending Aspect-Oriented Programming in Order to Flexibly Support Workflows. In Proceedings of the ICSE98 AOP Workshop, Kyoto, Japan, April 1998, S. 41 – 46

[ScAs98d] R. Schmidt, U. Assmann. Concepts For Developing Component-based Systems. In Proceedings of the ICSE98 Workshop on Component-Based Software-Engineering, Kyoto, Japan, April 1998, http://www.sei.cmu.edu/activities/cbs/icse98/papers/p12.html

[ScBö94] W. Schulze, M. Böhm: Klassifikation von Vorgangs- und Dokumentenverwaltungssystemen. In GI-Fachgruppentreffen EMISA/MOBIS 13./14. Oktober 1994, Münster, Oktober 1994.

[Schä96] T. Schäl: Workflow management systems for process organisations, LNCS 1096, Springer Verlag, Berlin, 1996.

[Sche94] A.-W. Scheer: Wirtschaftsinformatik. Referenzmodelle für industrielle Geschäftsprozesse, 4. Auflage. Springer Verlag, Berlin, 1994.

[Schm93] R. Schmidt: Ereignisbasierte Ablaufsteuerung von Problemlösezyklen, Diplomarbeit, Universität Erlangen-Nürnberg, 1993.

[Schm97a] R. Schmidt: Component-based systems, composite applications and workflow-management. In Proceedings Workshop Foundations of Component-Based Systems, Zürich, Schweiz, 26. September, 1997, S. 206 — 214.

[Schm97b] R. Schmidt: Komponentenbasierte Realisierung von Workflow-Management-Systemen in [JaBS97]. S. 355 — 364.

[Schm97c] Rainer Schmidt. Workflow-Unterstützung auf der Basis von ActiveX und Java-Beans. In H.J. Scheibl (Hrsg.): 7. Kolloquium Software-Entwicklung: Methoden, Werkzeuge, Erfahrungen'97. Technische Akademie Esslingen (TAE), Germany, September 1997, S. 849 — 860.

[Scho98] G. Scholz: Evolutionäre Abbildung SOM-modellierter Geschäftsprozesse auf das Workflow-Management-System IBM FlowMark. Diplomarbeit an der Universität Karlsruhe, 1998.

[Schu97a] W. Schulze: Implementierungstechniken für WfMS in OMA-konformen Architekturen. In [JaBS97].

[Schu97b] H. Schuster: Architektur von Workflow-Managment-Systemen. In [JaBS97].

[Schu97c] W. Schulze: "Fitting the Workflow Management Facility into the Object Management Architecture". OOPSLA'97, 3rd Workshop on Design and Implementation of Business Objects, Altanta/Georgia, 4 Oktober, 1997.

[Schu97d] W. Schulze: Anwendung von Implementierungstechniken für verteilte Workflow-Management-Systeme. In [JaBS97].

[Schu98] H. Schuster: Architektur verteilter Workflow-Management-Systeme. DISDBIS 50, Infix, Sankt Augustin, 1998.

[Schu99] W. Schulze: Ein Workflow-Management-Dienst für ein verteiltes Objektverwaltungssystem. Dissertation, TU Dresden 1999.

[ScLu95] K. Schwab, H. Ludwig: Ein ereignisbasierter Ansatz zur Integration von Workflow-Management-Systemen mit Groupware-Werkzeugen. In W. Augsburger, H. Ludwig, K. Schwab (Hrsg.): Koordinationsmethoden und -werkzeuge bei der computergestützten kooperativen Arbeit. Bamberger Beiträge zur Wirtschaftsinformatik Nr. 30, Otto-Friedrich-Universität, Bamberg, 1995.

[ScOS94] G. Scherrer, A. Oberweis, W. Stucky: ProMISE - a Process Model for Information System Evolution. In Proceedings Third Maghrebian Conference on Software Engineering and Artificial Intelligence, Rabat, Marokko, 1994, S. 27 — 36.

[ScSt98] R. Schätzle, W. Stucky: EventFlow: An Event-based Workflow Modeling Language. In Proceedings EDBT ´98 Workshop on Workflow Management Systems, Valencia, 1998.

[Sheth97] A. P. Sheth: From Contemporary Workflow Process Automation to Adaptive and Dynamic Work Activity Coordination and Collaboration. In Proceedings DEXA Workshop 1997, S. 24 — 27.

[Sieb99] R. Siebert: An Open Architecture For Adaptive Workflow Management Systems. To appear in Transactions of the SDPS: Journal of Integrated Design and Process Science. Society for Design and Process Science, Austin, Texas, 1999.

[SiMl96] P. Singh, M. A. Vouk: Scientific Workflows: Scientific Computing Meets Transactional Workflows. NSF Workshop on Workflow and Process Automation in Information Systems: State-of-the-Art and Future Directions. The State Botanical Garden of Georgia, Athens, 8. — 10. Mai, 1996.

[Sing96a] M. Singh: Synthesizing Distributed Constrained Events from Transactional Workflows. Technical Report, Department of Computer Science, North Carolina State University, 1996.

[Sing96b] M. Singh: Distributed Scheduling of Workflow Computations, Technical Report, Department of Computer Science, North Carolina State University, 1996.

[Sinz95a] E. J. Sinz: Kann das Geschäftsprozeßmodell der Unternehmung das unternehmensweite Datenschema ablösen ? In: Computerwoche 15. September1995, auch Bamberger Beiträge zur Wirtschaftsinformatik Nr. 33, Universität Bamberg.

[Sinz95b] E. J. Sinz: Serviceorientierung der Hochschulverwaltung und ihre Unterstützung durch workflow-orientierte Anwendungssysteme. In: Knop J. (ed.): European University Information Systems. In Proceedings EUNIS'95 Congress, Düsseldorf 6. - 8. November 1995, S. 225 – 237.

[Sinz98] E. J. Sinz: Modellierung betrieblicher Informationssysteme - Gegenstand, Anforderungen und Lösungsansätze. In: K. Pohl, A. Schürr, G. Vossen (Hrsg.): Proceedings Modellierung ´98, Angewandte Mathematik und Informatik, Universität Münster, Bericht Nr. 6/98-I, S. 27 - 28

[SJKB94] H. Schuster, S. Jablonski, T. Kirsche, C. Bußler: A Client/Server Architecture for Distributed Workflow Management Systems. In Proceedings of the 3rd Int. Conference Parallel and Distributed Information Systems (PDIS'94), Austin, September 1994 , S. 253 — 256.

[SKMW96] A. Sheth, K. Kochut, J. Miller, D. Worah, S. Das, C. Lin, D. Palaniswami, J. Lynch and I. Shevchenko. Supporting State-wide Immunization Tracking using Multi-Paradigm Workflow Technology. In Proceedings of the 22nd Intl. Conf. on Very Large Databases (VLDB96), September 1996.

[StOS93] W. Stucky, A. Oberweis, G. Scherrer: INCOME/STAR: Process model support for the development of information systems, in: J. Niedereichholz, W. Schuhmann (Hrsg.): Wirtschaftsinformatik - Beiträge zur modernen Unternehmensführung, Festschrift zum 60. Geburtstag von Franz Steffens, Campus-Verlag, 1993, S. 145 — 165.

[Szyp92] C. Szyperski: Import is not Inheritance - Why we need both: Modules and Classes. In Proceedings Sixth European Conference on Object_Oriented Programming (ECOOP'92), Utrecht, Niederlande. LNCS Nr. 615, Springer Verlag, Juni 1992, S. 19 — 32.

[Szyp97] C. Szyperski: Component Software - A Market on the Verge of Success. The Oberon Tribune, 2(1), 1997.

[Szyp98] C. Szyperski: Component Programming, Beyond Object-Oriented Programming. Addison Wesley, New York, 1998.

[Taiv96] A. Taivalsaari: On the Notion of Inheritance. ACM Computing Surveys, 28(3), September 1996.

[ToGD96] D. Tombros, A. Geppert, K. R. Dittrich: Design and Implementation of Process-Oriented Environments with Brokers and Services. In B. Freitag, C.B. Jones, C. Lengauer, H.-J. Scheck: Object Orientation with Parallelism and Persistence, Kluwer Academic Publishers, 1996.

[ToGD97] D. Tombros, A. Geppert, K. R. Dittrich: The BROKER/SERVICES MODEL for the DESIGN of Cooperative Process-Oriented Environments. Technical Report 97.06, Department of Computer Science, University of Zürich. 1997.

[Udel94] J. Udell: Componentware. Byte Magazine, 19(5), 1994, S. 45 — 56.

[UML] http://www.rational.com

[VoBe96] G. Vossen, J.Becker: (Hrsg.): Geschäftsprozeßmodellierung und Workflow-Management. Thomson Publishing, 1996.

[Voge98] A. Vogel: CORBA and Enterprise Java Beans-based Electronic Commerce. International Workshop on Component-based Electronic Commerce, Berkeley, Kalifornien, 25 Juli, 1998.

[VoWe98] G. Vossen, M. Weske: The WASA Approach to Workflow Management for Scientific Applications, in: A. Dogac, L. Kalinichenko, T. Özsu, A. Sheth (Hrsg.): Workflow Management Systems and Interoperability. NATO ASI Series F: Computer and System Sciences, Vol. 164, Springer Verlag, Berlin, 1998, S. 145 — 164.

[VoWe99] G. Vossen, M. Weske: The WASA2 Object-Oriented Workflow Management System. In Proceedings SIGMOD Conference 1999, S. 587 — 589.

[VWWi96] G. Vossen, M. Weske, G. Wittkowski: Prototypical Implementation of WASA: Flexible and Platform-Independent Workflow Management. Fachbericht Angewandte Mathematik und Informatik 23/96-I, Universität Münster, 1996.

[Wang95] X. Wang: Implementation and Evaluation of CORBA-Based Centralized Workflow Schedulers. Master's thesis. University of Georgia, August 1995.

[WCOP96] WCOP96 , Linz 1996. Workshop Reader of the 10th European Conference on Object-Oriented Programming ECOOP96. Dpunkt Verlag, Heidelberg, 1996

[WDMS95] D. Wodtke, A. Kotz Dittrich, P. Muth, M. Sinnwell, G. Weikum. MENTOR: Entwurf einer Workflow-Management-Umgebung basierend auf State- und Activitycharts. In: [Laus95]

[Weck96] W. Weck: Idenpendently Extensible Component Frameworks. In [WCOP96].

[Weich97] M. Weichhold: Abbildung eines SOM-modellierten Geschäftsprozesses auf Lotus Notes. Diplomarbeit an der Universität Karlsruhe, 1997.

[WeMW98] J. Weissenfels, P. Muth, G. Weikum: Flexible Worklist Management in a Light-Weight Workflow Management System. In Proceedings of EDBT Workshop on Workflow Management Systems, Valencia, 1998.

[Wesk96] M. Weske: Modellierung von Workflows. GI-AK-Workflow Treffen am 6.11.95 in Frankfurt. http://is2221.inf.tu-dresden.de/~www/GI-Dokumente/Weske_1.ps

[Wesk98] M. Weske: Object-Oriented Design of a Flexible Workflow Management System. In Proceedings ADBIS 1998, S. 119 — 130.

[WeSz96] W. Weck, C. Szyperski: Do We Need Inheritance? Workshop on Composability Issues in Object-Orientation (at ECOOP'96), Linz, Austria. June 1996.

[Wewe96] T. Wewers: Konzeption eines zwischenbetrieblichen Workflow-Management-Systems zur Unterstützung von Geschäftsprozessen bei der Sonderabfallentsorgung. Arbeitspapier Nr. 4/96 des Bereichs Wirtschaftsinformatik 1 der Universität Erlangen-Nürnberg, Nürnberg 1996.

[Wewe98] T. Wewers: Zwischenbetrieblich integriertes Workflow-Management, dargestellt am Beispiel von Geschäftsprozessen bei der Sonderabfallentsorgung. Dissertation an der Universität Erlangen-Nürnberg, Nürnberg, 1998

[WFFA] ftp://ftp.omg.org/pub/docs/bom/98-06-07.pdf

[WfMC] Workflow Management Coalition: http://www.aiai.ed.ac.uk/project/wfmc/

[WHKS98] M. Weske, J. Hündling, D. Kuropka, H. Schuschel: Objektorientierter Entwurf eines flexiblen Workflow-Management-Systems. Informatik Forschung und Entwicklung. 13. Jahrgang, Nr. 4, S. 179 — 195.

[WIDE] WIDE Projekt: Project Description and Objectives. http://www.sema.es/projects/WIDE/Objectives.html

[WKMS95] D. Wodtke, A. Kotz Dittrich, P. Muth, M. Sinnwell, G. Weikum: MENTOR: Entwurf einer Workflow-Management-Umgebung basierend auf State- und Activitycharts. In: [Laus95].

[Wodt96] D. Wodtke: Modellbildung und Architektur von verteilten Workflow-Management-Systemen. DISDBIS 31, infix Verlag, Sankt Augustin, 1996.

[WoWe97] D. Wodtke, G. Weikum: A Formal Foundation for Distributed Workflow Execution Based on State Charts. In Proceedings ICDT 1997, S. 230 — 246.

[Würg98] C. Würges: Workflow-Unterstützung durch komposite Anwendungen auf der Basis von JavaBeans. Diplomarbeit an der Unversität Karlsruhe, 1998.

[WVMe96] M. Weske, G. Vossen, C. B. Medeiros: Scientific Workflow Management: WASA Architecture and Applications. Fachbericht Angewandte Mathematik und Informatik 03/96-I, Universität Münster, 1996.

[WWDM97] G. Weikum, D. Wodtke, A. Kotz Dittrich, P. Muth, J. Weissenfels: Spezifikation und verteilte Ausführung von Workflows in MENTOR. In: Informatik Forschung und Entwicklung, Themenheft Workflow-Management, Januar 1997.

[WWVB96] J. Wainer, M. Weske, G. Vossen, C. Bauzer-Medeiros: Scientific workflow systems. NSF Workshop on Workflow and Process Automation in Information Systems: State-of-the-Art and Future Directions. The State Botanical Garden of Georgia, Athens, 8.-10. Mai, 1996.

[WWW] http://www.w3.org

[WWWK96] D. Wodtke, J. Weissenfels, G. Weikum, A. Kotz-Dittrich: The Mentor project: Steps towards enterprise-wide workflow management. In Proceedings 12th IEEE International Conference on Data Engineering, 1996.

[WWWK96] J. Weissenfels, D. Wodtke, G. Weikum, A. Kotz Dittrich: The MENTOR Architecture for Enterprise-wide Workflow Management. In: [Shet96].

[XML] http://www.w3.org

[Zimm97] W. Zimmer: Frameworks und Entwurfsmuster. Shaker Verlag, Aachen, 1997.