



Xpert.press

Joachim Müller

# Workflow-based Integration

 Springer

**Xpert.press**

Die Reihe **Xpert.press** vermittelt Professionals  
in den Bereichen Softwareentwicklung,  
Internettechnologie und IT-Management aktuell  
und kompetent relevantes Fachwissen über  
Technologien und Produkte zur Entwicklung  
und Anwendung moderner Informationstechnologien.

Joachim Müller

# Workflow-based Integration

Grundlagen, Technologien, Management

Mit 102 Abbildungen und 21 Tabellen

 Springer

Joachim Müller

jo.mueller@gmx.net

info@workflow-based-integration.de

www.workflow-based-integration.de

Bibliografische Information der Deutschen Bibliothek

Die Deutsche Bibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über <http://dnb.ddb.de> abrufbar.

ISSN 1439-5428

ISBN 3-540-20439-3 Springer Berlin Heidelberg New York

Dieses Werk ist urheberrechtlich geschützt. Die dadurch begründeten Rechte, insbesondere die der Übersetzung, des Nachdrucks, des Vortrags, der Entnahme von Abbildungen und Tabellen, der Funksendung, der Mikroverfilmung oder der Vervielfältigung auf anderen Wegen und der Speicherung in Datenverarbeitungsanlagen bleiben, auch bei nur auszugsweiser Verwertung, vorbehalten. Eine Vervielfältigung dieses Werkes oder von Teilen dieses Werkes ist auch im Einzelfall nur in den Grenzen der gesetzlichen Bestimmungen des Urheberrechtsgesetzes der Bundesrepublik Deutschland vom 9. September 1965 in der jeweils geltenden Fassung zulässig. Sie ist grundsätzlich vergütungspflichtig. Zuwiderhandlungen unterliegen den Strafbestimmungen des Urheberrechtsgesetzes.

Springer ist ein Unternehmen von Springer Science+Business Media

[springer.de](http://springer.de)

Haftungshinweis: Trotz sorgfältiger Prüfung übernehmen weder Springer noch der Autor eine Haftung für die Inhalte der in diesem Buch zitierten Internet-Seiten. Für den Inhalt der zitierten Seiten und auch der mit diesen Seiten wieder verlinkten Seiten sind ausschließlich deren Betreiber verantwortlich.

© Springer-Verlag Berlin Heidelberg 2005

Printed in Germany

Die Wiedergabe von Gebrauchsnamen, Handelsnamen, Warenbezeichnungen usw. in diesem Werk berechtigt auch ohne besondere Kennzeichnung nicht zu der Annahme, dass solche Namen im Sinne der Warenzeichen- und Markenschutzgesetzgebung als frei zu betrachten wären und daher von jedermann benutzt werden dürften. Text und Abbildungen wurden mit größter Sorgfalt erarbeitet. Verlag und Autor können jedoch für eventuell verbliebene fehlerhafte Angaben und deren Folgen weder eine juristische Verantwortung noch irgendeine Haftung übernehmen.

Umschlaggestaltung: KünkelLopka Werbeagentur, Heidelberg

Satz: durch den Autor unter Benutzung eines Springer Word-Makropakets

Herstellung: LE-TeX Jelonek, Schmidt & Vöckler GbR, Leipzig

Gedruckt auf säurefreiem Papier 33/3142/YL - 5 4 3 2 1 0

*Für Hendrik, Anneke und Andrea*

## Vorwort

Der Titel *Workflow-based Integration* sagt aus, dass der Ansatz für die Integration die fachliche Anforderung ist, also der Geschäftsprozess, der durch eine zu bestimmende Technologie unterstützt werden soll. Diese Unterstützung umfasst nicht nur einen begrenzten Bereich des Geschäftsprozesses, sondern den kompletten Ablauf, z.B. von der Anfrage für ein Darlehen über seine Auszahlung bis zur abgeschlossenen Tilgung. Während dieses Ablaufes werden unterschiedliche Ressourcen in Form von Mitarbeitern und Geräten (Hardware und Werkzeuge) benötigt und müssen für den reibungslosen Ablauf zur Verfügung gestellt werden.

Die Entwicklung von prozessunterstützenden Anwendungen wird immer komplexer und aufwändiger. Hinzu kommt, dass die Projektbeteiligten oft schon beim Projektstart unterschiedlicher Auffassung über Begriffe und deren Bedeutung sind. Dieses Buch soll dabei helfen, die Kommunikation zwischen den Projektbeteiligten zu verbessern, um gemeinsam die Ziele eines Projektes bestimmen zu können und diese zu erreichen.

Zu Beginn des Buches erhält der Leser Grundlagen über die Themen, die für eine erfolgreiche Durchführung prozessorientierter Projekte notwendig sind. Diese Grundlagen umfassen einen Einblick in die Begriffsklärung, die Modellierung, die Dokumentation und die Technologie, mit der solche Prozesse beschrieben und umgesetzt werden können. Weiterhin wird ein Ausblick gewagt, welche zukünftigen Technologien in Zusammenhang mit Workflow zu berücksichtigen sind. Ein Vorgehensmodell, welches im Anschluss vorgestellt wird, soll mit seinen Templates einen Leitfaden für die effektive Durchführung von Workflow-Projekten darstellen. Danach wird eine Auswahl von Werkzeugen mit unterschiedlichen Technologien vorgestellt, die dem Leser helfen soll, eine Entscheidung für ein Werkzeug zu treffen, welches die unternehmensspezifischen Anforderungen erfüllt. Unterstützt wird die Auswahl durch die Zusammenstellung von Auswahlkriterien, die für eine Evaluierung notwendig sind. Am Schluss des Buches findet der Leser eine Erklärung der Templates, die während des Projektverlaufs verwendet werden können.

Das Buch wendet sich vornehmlich an Entscheider, aber auch an Entwickler, die einen fundierten Einstieg in die Grundlagen, Technologien und Management-Techniken der „Workflow-based Integration“ sowie der „Enterprise Application Integration“ suchen. Dabei werden lediglich allgemeine Kenntnisse der IT-Technologie vorausgesetzt.

An dieser Stelle möchte ich mich auch für die Unterstützung derer bedanken, die den Text und die Abbildungen angepasst haben. Mein Dank richtet sich besonders an Frank Dammeyer, Holger Koschek und Michael Köhlmann. Ich möchte mich auch beim Springer-Verlag für die angenehme Zusammenarbeit bedanken, besonders bei Herrn Frank Schmidt, der mich während des ganzen Zeitraumes der Bucherstellung sehr gut unterstützt hat.

Lorsch, im Juli 2004

*Joachim Müller*



# Inhaltsverzeichnis

<b>Einleitung .....</b>	<b>1</b>
Überblick.....	5
Vorgezogenes Fazit.....	6
 <b>1     Workflow .....</b>	 <b>7</b>
1.1   Was ist ein Geschäftsprozess? .....	7
1.2   Was ist ein Workflow?.....	8
1.3   Was ist Workflow Management?.....	10
1.4   Workflow vs. Geschäftsprozess .....	14
1.4.1   Historie der Workflow-Management-Systeme .....	14
1.5   WfMC (Workflow Management Coalition).....	17
1.5.1   Workflow Reference Model .....	17
1.5.2   Metamodell eines Workflows.....	20
1.6   Softwarearchitektur einer workflow-basierten Lösung .....	21
1.7   Warum Workflow einführen? .....	24
1.7.1   Aktuelle Situation in den Unternehmen.....	24
1.7.2   Vorteile von workflow-basierten Lösungen .....	25
1.7.3   Prozesseigenschaften für den Einsatz eines WfMS .....	27
1.8   Der Workflow-Regelkreis.....	28
1.9   Kommentar .....	30
1.9.1   Prozessdefinition .....	31
1.9.2   Architektur.....	32
1.9.3   WfMC.....	33
 <b>2     Enterprise Application Integration (EAI).....</b>	 <b>35</b>
2.1   Begriffserklärung .....	35
2.1.1   Formen von EAI .....	36
2.1.2   Integrationsarten .....	38
2.2   Entwicklung von EAI .....	42
2.2.1   Innerhalb der Unternehmen .....	42
2.2.2   Zu anderen Unternehmen .....	44
2.3   Eigenschaften eines Integrationsservers .....	44
2.4   EAI-Architekturen .....	51
2.4.1   Point-to-Point.....	52
2.4.2   Hub & Spoke .....	53
2.4.3   Bus-oriented.....	53
2.4.4   Distributed Objects .....	54
2.5   Middleware .....	55
2.6   Kommentar .....	56

<b>3</b>	<b>Software-Architekturen.....</b>	<b>59</b>
3.1	Software-Architekturarten .....	64
3.1.1	Zwei-Schichten-Architektur .....	64
3.1.2	Drei-Schichten-Architektur .....	64
3.1.3	Vier-Schichten-Architektur .....	65
3.2	Java 2 Enterprise Edition (J2EE) .....	66
3.3	Weitere objektorientierte Architekturen .....	70
3.3.1	Verteilte Anwendungen mit CORBA .....	71
3.3.2	Verteilte Anwendungen mit Microsoft .NET .....	75
3.4	Kommentar .....	77
<b>4</b>	<b>Prozessmodellierung .....</b>	<b>79</b>
4.1	Modellierungsmethoden .....	86
4.1.1	Petrinetze .....	87
4.1.2	Ereignisgesteuerte Prozessketten (EPK) .....	94
4.1.3	Unified Modeling Language (UML) .....	99
4.1.4	Gegenüberstellung .....	112
4.2	Kommentar .....	113
<b>5</b>	<b>Workflow-based Integration (Wfbl) .....</b>	<b>115</b>
5.1	Modellierungsansatz .....	115
5.2	Technischer Ansatz .....	117
5.2.1	Variante 1 .....	118
5.2.2	Variante 2 .....	119
5.2.3	Client-Varianten .....	122
5.2.4	Authentifizierung und Autorisierung .....	124
5.3	Kommentar .....	125
<b>6</b>	<b>Workflow-Projekte .....</b>	<b>127</b>
6.1	Einleitung .....	127
6.2	Projektablauf/Vorgehensmodell .....	127
6.2.1	Initial .....	130
6.2.2	Analyse der bestehenden Prozesse .....	132
6.2.3	Projektziel bestimmen .....	133
6.2.4	Prozess modellieren .....	135
6.2.5	Feinmodellierung und technische Konzeption .....	139
6.2.6	Geschäftsprozess implementieren .....	148
6.2.7	Test des Geschäftsprozesses .....	149
6.2.8	Geschäftsprozess freigeben .....	150
6.2.9	Laufzeitdaten erfassen .....	150
6.2.10	Auswerten der Laufzeitdaten und Redesign des Prozesses .....	151
6.3	Projektorganisation und Projektrollen .....	152
<b>7</b>	<b>Ausblick .....</b>	<b>157</b>
7.1	Web Services .....	157
7.1.1	Definierte Rollen .....	158
7.1.2	Technik .....	158

7.1.3	Web Services in Verbindung mit Workflow .....	159
7.1.4	Kommentar .....	161
7.2	Open-Source .....	162
7.2.1	Was ist Open-Source?.....	162
7.2.2	Lizenzmodelle .....	163
7.2.3	Open-Source in Verbindung mit Workflow .....	164
7.2.4	Kommentar .....	166
<b>8</b>	<b>Werkzeuge .....</b>	<b>167</b>
8.1	Workflow-Management-Systeme .....	167
8.1.1	CARNOT.....	168
8.1.2	e-Work.....	176
8.1.3	EasyFlow .....	183
8.2	Auswahlkriterien für ein WfMS .....	190
8.3	Eigenentwicklung eines WfMS .....	195
8.4	Modellierungswerkzeuge .....	198
8.4.1	ADONIS .....	198
8.4.2	ARIS .....	199
8.5	Kommentar .....	200
<b>9</b>	<b>Template .....</b>	<b>203</b>
9.1	Projektziele bestimmen.....	203
9.2	Prozesslandkarte erstellen.....	205
9.3	Erfolgsfaktoren ermitteln .....	206
9.4	Prozesse bewerten.....	207
9.5	Projektbeteiligte ermitteln.....	208
9.6	Prozesse bewerten und priorisieren.....	209
9.7	Mengengerüst erfassen.....	211
9.8	Ressourcen erfassen .....	212
9.9	Verbesserungspotential bewerten .....	213
9.10	Teilprojekte identifizieren.....	215
9.11	Prozessschnittstellen aufnehmen.....	216
9.12	Identifizierte Anwendungen.....	217
<b>10</b>	<b>Anhang.....</b>	<b>219</b>
10.1	ARIS-Konzept .....	219
10.1.1	Phasen des HOBE.....	219
10.1.2	ARIS-Haus .....	220
10.1.3	Objekte in ARIS .....	221
10.1.4	Sichten in ARIS .....	221
10.2	WfMC-Dokumenten-Index .....	222
	<b>Glossar.....</b>	<b>225</b>
	<b>Literaturverzeichnis.....</b>	<b>233</b>
	<b>Stichwortverzeichnis .....</b>	<b>237</b>

# Einleitung

Die Anwendungsentwicklung, welche als übergreifender Begriff für die Erstellung von Lösungen mit Informationstechnologie(IT)-Unterstützung zu verstehen ist, muss sich dem Trend der Globalisierung und Fusion von Unternehmen anpassen. Dadurch entstehen immer kürzer werdende Zyklen für die Überarbeitung von realisierten IT-Lösungen, die nicht selten mit hohem Aufwand angepasst werden müssen.

Aufgrund immer kürzerer Produkt- und Innovationszyklen müssen Unternehmen am Markt effektiver und flexibler reagieren. Dies erfordert eine gute Strukturierung der Geschäftsprozesse in den Unternehmen, damit sie auch ohne großen personellen Aufwand angepasst werden können. Die Einführung eines Workflow-Management-Systems (WfMS) kann diese Anforderungen auf der Basis von projektnotwendigen „Vorarbeiten“, die vor dessen Realisierung durchgeführt werden müssen, positiv unterstützen. Unter den Vorarbeiten versteht man die Analyse und Dokumentation der Prozesse, deren IT-Unterstützung und Automatisierung das Ergebnis für das Unternehmen optimiert und somit die Einführung eines WfMS zum Ziel hat.

Das vorliegende Buch soll aber nicht die Einführung eines WfMS als Allheilmittel für die Minimierung der Projektaufwände und der effektiven Realisierung von automatisierten Geschäftsprozessen darstellen. Es soll vielmehr einen Überblick über die prozessbasierte Integration von Anwendungen geben. Dies muss nicht zwingend die Einführung eines WfMS zur Folge haben, oft genügt die Optimierung der Geschäftsprozesse. Man kann nicht pauschal sagen, dass es immer sinnvoll ist, einen Geschäftsprozess mit einem WfMS zu automatisieren. Es ist aber ein mögliches Ergebnis einer entsprechenden Analyse, welche die Geschäftsprozesse in entsprechende Automatisierungskategorien einteilt. Diese Einteilung der Geschäftsprozesse wird in den Kapiteln Workflow und Workflow-Projekte näher erläutert.

Unter den Begriffen Workflow und WfMS wird sehr vieles zusammengefasst und es gibt es sehr viel und Unterschiedliches zu lesen – insbesondere von Produktherstellern, die einen expandierenden Markt für Workflow und somit für WfMS-Produkte sehen. Allerdings gibt es nicht viele Produkthersteller, die sich aus ihrer Historie heraus ausreichend mit diesem Thema auseinander gesetzt haben. Fast alle Software-Produkthersteller, die das Thema Workflow durch ihr Produkt abdecken wollen, haben eine unterschiedliche Historie und zum Teil auch ein anderes Verständnis von Workflow. Ein solcher historischer Ursprung kann z.B. ein Dokumenten-Management-System (DMS), ein E-Mail-System, ein Integrationsserver

oder eine Office-Suite sein. Je nach Ursprung des Produktes sind Prozesssteuerungsfunktionalitäten gut oder weniger gut ausgeprägt. Der Kunde hat, wie so oft, die Qual der Wahl. Nicht selten fehlen ihm messbare Entscheidungskriterien für die Produktauswahl.

Die Gründung der WfMC (Workflow Management Coalition) wirkte sich in diesem Fall positiv auf die Entwicklung der Produkte aus, da die WfMC durch ihre definierte Architektur und Rollenbeschreibung ein idealisiertes WfMS vorgibt. Dieses kann als Anhaltspunkt für die Unterscheidung und Bewertung von WfMS-Produkten verwendet werden. Allerdings ist auch zu berücksichtigen, dass die WfMC aufgrund der Vielzahl ihrer Mitglieder mit unterschiedlichsten Interessen nur einen kleinsten gemeinsamen Nenner vorgegeben hat.

Seit ca. zehn Jahren findet man Workflow-Management-Systeme, die durch ihre Entwicklung inzwischen fast allen prozesstechnischen Anforderungen gewachsen sind. Diese unterstützen nicht nur die Automatisierung von Prozessen, sondern weisen auch einen hohen Grad an Anbindungs- und Integrationsmöglichkeiten auf, um die benötigten und/oder bereits vorhandenen Fachanwendungen anbinden zu können. Eine solche Integration kann so weit gehen, dass die eigentliche Fachanwendung für den Sachbearbeiter nicht mehr sichtbar ist und der zuvor modellierte Prozess vollständig vom WfMS gesteuert wird.

In dem vorliegenden Buch soll das Thema prozessorientierte Integration nicht wissenschaftlich erklärt werden. Der Leser wird vielmehr anhand von praktikablen und verständlichen Erklärungen an das Thema herangeführt. In den folgenden Kapiteln werden die Unterschiede zwischen den Begriffen Workflow und Geschäftsprozess erklärt, um die Schwerpunkte des anzuwendenden Vorgehensmodells besser verständlich und nachvollziehbar zu machen.

Nach Meinung des Autors gibt es während der Durchführung von Workflow-Projekten zwei Schwerpunkte. Dies sind zum einen der technische und zum anderen der fachliche Schwerpunkt. Daraus ergibt sich eine unterschiedliche Besetzung (Rollen) des Projektteams und die Gestaltung der Themeninhalte der Teilprojekte. Die Einbeziehung von unterschiedlichen Rollen aus dem Unternehmen zur Identifizierung, Definition und anschließenden Realisierung eines Prozesses ist für eine erfolgreiche Durchführung von Workflow-Projekten von erheblicher Bedeutung. So ist die fachliche Seite mit Personen aus dem Management und den Fachabteilungen zu besetzen, die aufgrund ihrer Aufgaben und Erfahrungen die Geschäftsprozesse gestalten und mit ihnen arbeiten oder arbeiten werden. Die fachliche Seite hat die Aufgabe, die Unternehmensentwicklung zielorientiert bis zur Umsetzung der Hilfsprozesse voranzutreiben. Der Fokus liegt hier also auf dem Geschäftsprozess und der Kenntnis, diese Geschäftsprozesse zu definieren und zu dokumentieren. Auf der technischen Seite werden die IT-Architekturen und die programmiertechnische Entwicklung auf der Basis der Geschäftsprozessdefinition realisiert. Die modellierten Prozesse müssen unter Zuhilfenahme der bestehenden Anwendungen des Unternehmens und ggf. unter Verwendung eines WfMS realisiert werden.

Das vorliegende Buch baut dem Leser die wichtige Brücke zwischen der fachlichen Modellierung und der technischen Realisierung der Geschäftsprozesse. Die Erfahrung hat gezeigt, dass neben der Fähigkeit, die beiden beteiligten Seiten zu verstehen, die Kommunikation zwischen den Projektbeteiligten mit das Wichtigste in einem Projekt ist. Die Kommunikation ist aufgrund der unterschiedlichen Schwerpunkte und Wissensstände der Beteiligten in der Regel nicht einfach. So haben gleiche Begriffe u.U. in den differenzierten Bereichen (z.B. Sachbearbeiter Finanzdienstleistung und IT-Spezialist) oft eine andere Bedeutung, oder die Beteiligten gehen aufgrund unterschiedlicher Sichtweisen von einer anderen Bedeutung aus. Das Etablieren einer funktionierenden und effektiven Kommunikation zwischen einem Sachbearbeiter, dem Software-Architekten und dem SW-Realisierer ist somit die größte Herausforderung für den Projektleiter. Ziel muss es sein, dass die Projektbeteiligten denselben Sprachgebrauch verwenden und ein gemeinsames Verständnis bezüglich des Vorgehens und der technischen Realisierung haben. Dieses Buch beschreibt alle Phasen eines Workflow-Projektes und stellt ein Vorgehensmodell vor, welches als Leitfaden in Projekten verwendet werden kann. Auf eine zu starke Detaillierung der Thematik wird zugunsten einer besseren Verständlichkeit und einfacheren Lesbarkeit verzichtet. Ziel ist es, einen Überblick über die Einführung einer Workflow-gestützten Lösung (Geschäftsprozess) zu verschaffen und dem Leser die Möglichkeit zu geben, ein Workflow-Projekt korrekt zu starten und im Projektverlauf Entscheidungen treffen zu können, die ein erfolgreiches Projektende gewährleisten.

Der Projekterfolg ist nicht zuletzt notwendig, um die Kosten in den Fachabteilungen und auch in den IT-Abteilungen zu optimieren. Dadurch ist ein effektiveres Vorgehen in den Projekten mehr denn je gefragt. Workflow-Projekte bieten an dieser Stelle den Vorteil, dass sie mit unterschiedlichen Motivationen gestartet werden können und das Projektziel in (kleinen) fachlich abgeschlossenen Schritten erreicht werden kann. Somit ist es möglich, die Kosten überschaubar und in einem sehr begrenzten Rahmen zu halten und die Anwendung in sukzessiven Schritten zu realisieren. Die Gründe für die unterschiedlichen Projektmotivationen in den Unternehmen ergeben sich aus der Ausstattung der IT-Infrastruktur, den Prozessdokumentationen und der vorhandenen Applikationslandschaft. Ein Teilprojekt kann darin bestehen, die Geschäftsprozesse zu modellieren und zu dokumentieren. In diesem Fall benötigt man einen fachlich erfahrenen Sachbearbeiter und einen (häufig externen<sup>1</sup>) Mitarbeiter, der Erfahrung in der Modellierung von Prozessen hat. Hinzu kommt ein Werkzeug, welches die modellierten Prozesse dokumentiert. Eine weiteres Teilprojekt kann auf der Basis von bereits dokumentierten und etablierten Geschäftsprozessen und einer heterogenen Software-Produktlandschaft initiiert werden. In diesem Fall ist festzustellen, welche Geschäftsprozesse am geeignetsten sind, um durch ein WfMS unterstützt zu werden. Es können so schnelle Erfolge (Quick-Wins) erzielt werden, die bereits die entstandenen Projektkosten rechtfertigen.

---

<sup>1</sup> Extern deswegen, weil ein externer Mitarbeiter einen unabhängigen Blick auf die Abläufe im Unternehmen hat.

Es wird an dieser Stelle mit Absicht nicht von einem ROI (Return On Investment) gesprochen, da dieser aufgrund häufig nicht vorhandener bzw. unvollständiger Ist-Daten nicht korrekt berechnet werden kann. Diese Ist-Daten beziehen sich beispielsweise auf die Prozesslaufzeit und die daraus entstandenen Prozesskosten. Eine optimale Ausgangssituation liegt vor, wenn die Geschäftsprozesse modelliert und mit einem Modellierungswerkzeug bereits dokumentiert sind, die Lizenzen eines WfMS im Unternehmen vorhanden sind und im Unternehmen eine heterogene Software-Produktlandschaft besteht. Ist dies alles der Fall, kann schnell mit einer technischen Realisierung der Geschäftsprozesse begonnen werden und die Projektkosten entstehen nur noch für die Umsetzung, also die IT-Unterstützung bzw. Automatisierung des Geschäftsprozesses.

Sind die fachlichen Rahmenbedingungen abgeklärt, kann mit der Zusammenstellung des Projektteams begonnen werden. Über den Bedarf an Projektbeteiligten kann man an dieser Stelle noch keine Aussage treffen, da das Ziel des Projektes noch nicht definiert wurde. Eine solche Zieldefinition wird leider sehr selten vor einem Projektstart durchgeführt und kann so die Ursache für unvorhergesehene Kosten sein. Es ist zwingend notwendig, eine Zieldefinition mit den Vertretern aller beteiligten Rollen aus den entsprechenden Projektphasen durchzuführen. Dies betrifft das Management, die Realisierer und die beteiligten Mitarbeiter, die diese Lösung verwenden sollen.

Aus den oben aufgeführten Beispielen der Projektmotivationen ist ersichtlich, dass Workflow-Projekte aus unterschiedlichen Ausgangssituationen initiiert werden können und trotzdem zum Erfolg führen. Es ist aber darauf zu achten, dass das Projekt in entsprechend kleine Realisierungsphasen aufgeteilt wird und dass diese Phasen am Ende Ergebnisse vorweisen können, die für den produktiven Einsatz des Geschäftsprozesses verwendet werden können. Es sind dabei folgende Punkte zu beachten:

- Vor dem Start eines Workflow-Projektes müssen die Abläufe des Unternehmens analysiert werden, um die vorliegende Ausgangssituation bestimmen zu können. Wichtig ist dabei, eine abteilungsübergreifende Analyse des betroffenen Geschäftsprozesses durchzuführen.
- Nicht jeder Prozess eignet sich für eine weitreichende Unterstützung/Automatisierung durch ein WfMS. Dies muss durch eine Analyse der Geschäftsprozesse geprüft werden.
- Workflow-Projekte müssen abteilungsübergreifend betrachtet werden, d.h. die Konsequenzen und Abhängigkeiten des Projektes und die daraus resultierenden Geschäftsprozesse müssen unternehmensweit berücksichtigt werden. Die Realisierung kann aber im ersten Schritt abteilungsbezogen erfolgen.

Dies soll vorerst an Ausführungen genügen. Die Details folgen in den einzelnen Kapiteln.

## Überblick

In den ersten Kapiteln Workflow, Enterprise Application Integration (EAI), und Software-Architekturen werden die Grundlagen der WfMS und des Workflow-Managements erläutert. Es werden neben den technischen Eigenschaften von WfMS auch die eines Integrationservers und der allgemeinen Softwarearchitektur erklärt. Diese Kapitel haben einen eher technischen Schwerpunkt und tragen primär dazu bei, dem Manager und den Sachbearbeitern des Fachbereichs das Thema Workflow und dessen technische Realisierung näher zu bringen.

Das darauf folgende Kapitel Prozessmodellierung beschreibt die Modellierung der Prozesse mit dem Fokus auf der Umsetzung bzw. Automatisierung von Geschäftsprozessen. Mit diesem Kapitel werden die Realisierer und Betreiber angesprochen, um ein besseres Verständnis für ein Vorgehenskonzept in ihrem Unternehmen zu bekommen.

In dem Kapitel Workflow-based Integration (WfBI) werden die fachlichen und technischen Anforderungen einer IT-Umgebung beschrieben, um Geschäftsprozesse anhand des prozessorientierten Ansatzes zu realisieren. Dieses Kapitel ist gleichermaßen für die Fachexperten als auch für die technisch verantwortlichen Projektbeteiligten geschrieben.

Das Kapitel Workflow-Projekte ist das Schlüsselkapitel dieses Buches. Mit ihm soll der Leser in die Lage versetzt werden, ein Workflow-Projekt in einzelnen Schritten durchführen zu können. Neben einer detaillierten Beschreibung der einzelnen Schritte werden dem Leser Templates zur Verfügung gestellt. Sie ermöglichen eine bessere Kontrolle über die benötigten Aktivitäten. Diese Templates stehen unter [www.workflow-based-integration.de](http://www.workflow-based-integration.de) zum Download zur Verfügung und können unter Verwendung der Quellenangaben frei verwendet werden. Das Vorgehensmodell in diesem Kapitel beschreibt die Durchführung eines Workflow-Projektes von der Ausarbeitung der globalen Ziele über die Modellierung und Implementierung bis hin zur Auswertung der Laufzeitdaten und dem eventuell daraus folgenden Redesign eines Geschäftsprozesses.

In dem Kapitel Ausblick werden die zukünftigen technischen Möglichkeiten beschrieben, die zum einen durch neue Technologien und zum anderen durch neue Produkte zu erwarten sind.

Am Ende des Buches erfolgt die Vorstellung einer kleinen Auswahl von WfMS-Produkten, die auf unterschiedlichen Technologien basieren. Sie geben dem Leser die Möglichkeit, die Unterschiede zwischen weiteren Produkten selbstständig herauszuarbeiten und so eine Produktauswahl für sein Unternehmen treffen zu können. Weiterhin findet der Leser im Anhang eine Kurzbeschreibung des ARIS-Konzeptes und der UML 2.0, mit der die Projektphasen von der Modellierung bis zur Programmierung dokumentiert werden können.



## Vorgezogenes Fazit

Die folgende Aufzählung beschreibt die wichtigsten Schlussfolgerungen dieses Buches.

- WfMS sind nicht mit Applikationen zu verwechseln, die eine starre Ablaufsteuerung implementiert haben.
- Die Architektur eines WfMS kann nicht vollständig mit der WfMC übereinstimmen, da noch nicht alle Interfaces verabschiedet wurden.
- Es empfiehlt sich, nie ein Großprojekt zu starten, sondern immer kleine, durchdachte und überschaubare Projektschritte zu planen und durchzuführen.
- Das Einbeziehen der fachlichen und technischen Abteilungen muss ein vorrangiges Bestreben sein.
- Ein WfMS ist nicht mit einem DMS (Dokumenten-Management-System) gleichzusetzen. Ein DMS kann eine Ablaufsteuerung enthalten, dieser Ablauf ist aber auf das Verwalten der Dokumente beschränkt.

Aus Workflow und EAI wird WfBI (Workflow-based Integration). Mit anderen Worten bedeutet dies, dass dieses Buch den Ansatz verfolgt, eine Lösung basierend auf dem Geschäftsprozess zu realisieren und nicht auf Basis einer allgemeinen Anwendung wie einem ERP oder Ähnlichem. Dies ist nach Meinung des Autors wichtig, da bis auf zertifizierte Prozesse alle Prozesse individuell auf das Unternehmen und die Unternehmensziele angepasst werden müssen.

# 1 Workflow

In diesem Kapitel wird eines der beiden Hauptthemen des Buches beschrieben. Neben den Begriffserklärungen und der Beschreibung der verschiedenen Workflow-Typen erhält der Leser einen allgemeinen Überblick über das Thema Workflow. Hierzu zählen neben der Definition die Historie und der allgemeine Aufbau von Workflow-Management Systemen-(WfMS).

Da zu den folgenden Begriffen viele Definitionen bereits erstellt wurden, soll an dieser Stelle keine neue Erklärung abgegeben werden, sondern die bereits bestehenden in Verbindung mit der Definition der Workflow Management Coalition (WfMC) beschrieben werden. Die Begriffe und die Zusammenhänge zwischen Geschäftsprozess, Workflow, Workflow-Management und Workflow-Management-System werden erklärt. Danach wird noch einmal detailliert auf den Unterschied zwischen den Begriffen Geschäftsprozess und Workflow eingegangen.

## 1.1 Was ist ein Geschäftsprozess?

Beispiele für einen Geschäftsprozess sind „einen Kredit aufnehmen“ in einer Bank, „einen Bauantrag einreichen“ in einer Behörde oder die Entwicklung einer Anwendung in einem Software-Haus. Es handelt sich um einen Geschäftsvorfall in einer Wirtschaftseinheit mit einem definierten Anfang und einem definierten Ende, der jeweils zwischen Start und Ende einen unterschiedlichen Verlauf nehmen kann und dessen unterschiedliche fachliche Funktionen ausgeführt werden. Dieser Verlauf, mit dem Leistungen oder Informationen transportiert werden, wird zuvor durch die Modellierung des Geschäftsprozesses festgelegt. Ein Geschäftsprozess wird durch ein Ereignis initiiert und ist eine Folge von Aktivitäten, die in einer logischen Verbindung stehen. Der Ablauf des Geschäftsprozesses wird durch die Eingabe (Input) von erforderlichen Daten sowie internen und externen Ereignissen beeinflusst. Diese Eingaben erfolgen durch die den Aktivitäten über Rollen zugewiesenen Akteure und können hierarchie- und standortübergreifend sein. Außer der Eingabe der Daten oder Zustände durch die zugewiesenen Ressourcen (Rollen) besteht auch die Möglichkeit, dass diese Eingabe durch Drittsysteme erfolgt. Aufgrund der Eingabe der Daten erfolgt eine Reaktion des WfMS und die Eingabe wird ver- oder bearbeitet. Das Ergebnis dieser Aktionen ist die Ausgabe (Output) einer Aktivität, die für weitere Aktivitäten zur Verfügung steht.

Ein Geschäftsprozess ist immer auf ein Unternehmensziel hin ausgerichtet und ein Teil der Wertschöpfungskette des Unternehmens. Er kann mehrere Zulieferpro-

zesse und unterstützende Prozesse in seinem direkten Umfeld haben. Eine genauere Einteilung und Positionierung des Begriffes ist im Kapitel Prozessmodellierung zu finden.

## 1.2 Was ist ein Workflow?

Ein Geschäftsprozess wird technisch durch einen Workflow unterstützt. Der Geschäftsprozess kann mit seinem Arbeitsablauf in seiner Gesamtheit oder auch nur in Teilen unterstützt werden. Das heißt, die in dem Geschäftsprozess modellierten Aktivitäten müssen nicht zwangsläufig durch ein WfMS unterstützt werden, sondern können auch organisatorisch, mit nur vereinzelt eingesetzten Software-Werkzeugen gelöst sein. Ein Workflow läuft immer wieder nach demselben oder zumindest nach einem ähnlichen Schema ab. Der Ablauf wird durch Ereignisse beeinflusst, die den Start eines Geschäftsprozesses und damit des speziellen Workflows auslösen und die enthaltenen Aktivitäten bereitstellen und beenden können. Nach einem erfolgreichen Beenden der Aktivitäten oder einem Abbruch einer Aktivität muss der Workflow immer zu einem definierten Zustand kommen. Die Aktivitäten werden von unterschiedlichen Rollen ausgeführt, denen die benötigten Werkzeuge (Softwaretools) und Informationen für das Ausführen der Aktivitäten zur Verfügung gestellt werden müssen.

Das internationale Standardisierungsgremium, die Workflow Management Coalition, unterscheidet vier grundlegende Workflow-Typen:

- Der *Ad-hoc-Workflow* unterstützt einmalige oder stark variierende Prozesse, die wenig strukturiert und nicht vorhersehbar sind.
- Der *Collaborative Workflow* unterstützt das gemeinsame Erarbeiten eines Ergebnisses; dieser Begriff wird auch als Synonym für Groupware verwendet.
- Der *Administrative Workflow* unterstützt strukturierte Routineabläufe, die nicht strategisch, selten zeitkritisch und von geringem Geldwert sind.
- Der *Production Workflow* unterstützt fest strukturierte und vordefinierbare Vorgänge, die zumeist zeitkritisch und von strategischer Bedeutung sind.

Eine andere Einteilung der Workflow-Typen verwendet Picot (Picot u. Rohrbach 1995). Nach Picot werden für eine genauere Bestimmung der Workflow-Typen fünf Kriterien verwendet. Diese Kriterien nennt Picot Prozess-Variablen. Sie sind in Tabelle 1.1 mit den Indikatoren aus den Geschäftsprozessen dargestellt.

**Tabelle 1.1** Prozessvariablen nach Picot

Prozessvariable	Indikatoren
Komplexität	Zahl der Teilaufgaben Anordnung der Teilaufgaben (sequentiell, parallel) Abhängigkeiten/Rückkoppelungsbedarf der Teilaufgaben Rollen der in den Prozess involvierten Mitarbeiter
Grad der Veränderlichkeit	Wiederholungshäufigkeit ohne Strukturveränderungen Planbarkeit der Kommunikation während der Informationsbeschaffung Offenheit des Prozessergebnisses Änderungsanfälligkeit, bedingt durch organisationsinterne/-externe Anforderungen
Detaillierungsgrad	Möglichkeit der Zerlegung des Gesamtprozesses in einfache Teilschritte Eindeutigkeit des erforderlichen Inputs, der Transformati-onsschritte und des Outputs
Grad der Arbeitsteilung	Anzahl der am Prozess beteiligten Mitarbeiter Koordinationsbedarf des Gesamtprozesses
Interprozessverflechtung	Schnittstellen zu anderen Prozessen Gemeinsame Datennutzung der Prozesse Prozesshierarchie (Beitrag des Prozesses zu über-, unter- oder nebengeordneten Prozessen)

Aus diesen Prozessvariablen ergeben sich nach Picot die drei Prozesstypen Routineprozess, Regelprozess und einmaliger Prozess. Diese werden anhand der Kombination der Prozessvariablen eingestuft. Picot detailliert die Prozesstypen noch genauer in entsprechende Teilprozesse. An dieser Stelle soll aber nicht näher darauf eingegangen werden.

### **Routineprozess**

- gut erkennbare Struktur
- beständig in der Struktur und somit über längere Zeit planbar
- standardisierter Ablauf
- hoher Grad der Arbeitsteilung
- geringe Anzahl der Schnittstellen zu anderen Prozessen

### **Regelprozess**

- kontrollierbare Struktur und Komplexität
- häufige, individuelle Veränderungen der Struktur durch Sachbearbeiter
- Abläufe eines Prozesstyps nicht determiniert
- individuelle Prozesse in der Regel bestimmbar

**Einmaliger Prozess**

- weder Ablauf noch beteiligte Rollen sind definierbar
- Ablauf durch einen Sachbearbeiter
- Prozess wird individuell bearbeitet
- keine Automatisierung der Prozesse sinnvoll

Aufgrund dieser Kombinationen kann folgende Einteilung (vgl. Abbildung 1.1) erstellt werden: mit ihr wird dargestellt, welcher Prozesstyp sich am besten eignet, um durch ein Workflow-Management-System unterstützt zu werden. Die Einteilung nach Picot sieht drei Bereiche, von geeignet bis ungeeignet, vor. Geeignet sind die Prozesse, in denen der Kernprozess und die Hilfsprozesse komplett standardisiert sind und einer hohen Wiederholungsrate unterliegen. Als ungeeignet zählen die Prozesse, die keinem planbaren Ablauf unterliegen und selten auftreten. Hier kann ein WfMS lediglich eine Unterstützung in Form einer vom Sachbearbeiter zu erstellenden Checkliste wahrnehmen. In dem Bereich dazwischen hat der Sachbearbeiter einen hohen Einfluss auf den Ablauf des Prozesses und verwendet neben einer eventuellen Prozessunterstützung in Form eines WfMS zusätzlich noch weitere Werkzeuge wie einen E-Mail-Client oder ein Dokumenten-Management-System. Oder es sind DV-technisch nicht unterstützbare Aktivitäten integriert, beispielsweise das Beschriften und Einlagern einer Baustoffprobe.

<div>Teilaufgaben im Proz.</div> <div>Prozesstyp</div>	Einzelfall- Aufgabe	sachbe- zogene Aufgabe	Routine- aufgabe
einmaliger Prozess	Checklisten -WF		
Regelprozess			
Routineprozess			WfMS

Abbildung 1.1. Eignung eines WfMS

**1.3 Was ist Workflow Management?**

Das Workflow Management befasst sich mit allen Aufgaben, die bei der Analyse, der Modellierung, der Simulation, der Reorganisation sowie bei der Ausführung und Steuerung von Workflows benötigt werden. Es stellt somit die einzelnen organisatorischen Arbeitsschritte und Abläufe zur Verfügung, die einem Lebenszyklus (Lifecycle) eines Workflow entsprechen, unabhängig davon, ob für die Unterstützung der Arbeitsschritte (Arbeitsphasen) ein Werkzeug in Form eines Programms

verwendet wird oder nicht. Das Workflow-Management-System ist ein System, welches die Phasen des Prozess-Lifecycles, also das Workflow-Management, durch IT-Werkzeuge unterstützt. Die Werkzeuge werden in Form von Software ausgeliefert und enthalten Komponenten für die Analyse, die Modellierung, die Steuerung, die Administration, die Simulation und das Monitoring von Workflows (s.u.). Häufig wird ein WfMS auch prozessorientierte Software genannt, da es sich wiederholende Abläufe nach einem zuvor festgelegten Schema steuert. Dadurch eignen sie sich besonders für Abläufe, die einen hohen Wiederholungs-, Standardisierungs- und Arbeitsteilungsgrad vorweisen können.

Das WfMS kann in verschiedene Software-Komponenten mit unterschiedlichen Aufgaben unterteilt sein. Diese können durch einen einzelnen Hersteller abgedeckt werden oder, falls eine Kooperation oder eine standardisierte Schnittstelle zwischen den Komponenten besteht, durch mehrere Hersteller. Werden Komponenten von unterschiedlichen Herstellern eingesetzt, so muss darauf geachtet werden, dass die einwandfreie Kommunikation über die Schnittstellen der Komponenten gewährleistet wird. Einen positiven Einfluss auf die „Zusammenarbeit“ der Komponenten von unterschiedlichen Herstellern will die WfMC haben. Die Ziele und die Standards der WfMC werden später noch beschrieben. Im Folgenden werden die Komponenten eines WfMS aufgeführt.

### **Modellierungskomponente**

Die Modellierungskomponente dient in erster Linie der grafischen Beschreibung der Prozesse. Ein Prozess muss so beschrieben werden, dass er von der Steuerungskomponente interpretiert und ausgeführt werden kann. Erst in zweiter Linie findet eine Abbildung der Organisation statt. Dieses Vorgehen unterstützt die Prozessbeschreibung, da die in den Prozessen beschriebenen Aktivitäten von über Rollen zugeordneten Akteuren (z.B. Sachbearbeiter, Abteilungsleiter) ausgeführt werden. In der Regel liegt die Organisation in einer Organisationsdatenbank, die ausgelesen werden kann. Ist dies nicht der Fall, muss die Organisation mit der Modellierungskomponente abgebildet werden. Eine Organisationsdatenbank bildet die Aufbauorganisation des Unternehmens ab und stellt die Verknüpfung zwischen den Mitarbeitern (oder abstrakt „Akteuren“) und den definierten Rollen her. Der Begriff Akteur ist allgemein gehalten, da dieser z.B. ein einfaches Programm oder ein Prozessautomat sein kann. Es ist dabei zu berücksichtigen, dass ein Akteur durchaus mehrere Rollen innehaben kann.

In dem aktuellen Produktangebot findet eine Überschneidung der angebotenen Modellierungswerkzeuge statt. Es sollte darauf geachtet werden, dass der WfMS-Hersteller einen grafischen Editor mitliefert, in dem die Prozesse in Verbindung mit der Steuerungskomponente definiert werden. Diese Komponenten werden auch als Definitionswerkzeuge bezeichnet, da sie einen anderen Schwerpunkt und damit geringen Leistungsumfang haben als die Modellierungswerkzeuge, die sich auf das Geschäftsprozessmanagement spezialisieren und somit den Fokus neben der Modellierung auch auf Simulation und Reorganisation legen (z.B. ARIS, ADONIS oder Aeneis).

### Steuerungskomponente

Wie unten noch näher beschrieben wird, ist der Kern eines WfMS die Steuerungskomponente (Workflow-Engine). Die Steuerungskomponente liest die Prozessdefinition und bildet daraus eine Prozessinstanz (z.B. eine Eingangsrechnung). Diese wird in Form eines Geschäftsvorfalles gestartet, gesteuert und protokolliert. Die Steuerungskomponente enthält Funktionen wie Terminüberwachung, Eskalationsmanagement, Wiedervorlage, Protokollierung, Starten eines Falles sowie Bereitstellen der Informationen und der Werkzeuge (z.B. Textverarbeitung) für die Bearbeitung des Geschäftsvorfalles bei den zuständigen Akteuren. Der Geschäftsvorfall (oder kurz Fall) wird dem Akteur in einem Pool, der in der Regel *Eingangskorb*, *Tätigkeitsliste* oder *offene Tasks* genannt wird, zur Verfügung gestellt. Dieser hat anhand des Workflow-Clients die Möglichkeit, den Pool auszulesen und so die Fälle zu bearbeiten. Da der Funktionsumfang einer Steuerungskomponente nicht standardisiert ist, unterscheiden sich auch diese WfMS-spezifischen Clients in ihrem Funktionsumfang.

### Überwachungskomponente

Die Überwachungskomponente ermöglicht es, zwei Arten von Informationen darzustellen. Die *instanzbezogenen* Daten werden durch die Steuerungskomponente protokolliert und dargestellt. Diese aufgezeichneten Daten ermöglichen es, das Laufzeitverhalten zu einem bestimmten Fall (Prozessinstanz) auszuwerten. Diese sollten den ausführenden Akteur, die Liegezeit, die Bearbeitungszeit und die Transportzeit beinhalten. Idealerweise werden vom WfMS auch fachliche Daten protokolliert, so dass in der Prozesshistorie der gesamte Geschäftsvorfall mit allen relevanten Daten für die spätere Nachvollziehbarkeit abgelegt ist. Damit werden die Anforderungen aus den Bereichen Revisionssicherheit und ISO 9000-Zertifizierung abgedeckt. Der Prozessinhaber und/oder die Führungskraft erhalten so eine bessere Steuerungsmöglichkeit, indem sie vollständig über den aktuellen Bearbeitungsstatus informiert werden und, falls notwendig, eingreifen können. Die zweite Art der Daten sind die (aggregierten) *vorgangsbezogenen Daten*. Diese ermöglichen es, eine Aussage darüber zu treffen, ob der Geschäftsprozessablauf im Mittel effektiv ist oder ob sich einzelne Aktivitäten als Flaschenhälse darstellen. Sie bilden die Grundlage für die permanente Reorganisations- und Optimierungsmöglichkeit des Vorganges.

### Schnittstellenkomponente

Die Architektur und Technologie der Schnittstellenkomponente ist mit die wichtigste Komponente des WfMS, da sie die Anbindung zur „Außenwelt“ ermöglicht. Für ein WfMS ist die Anbindung von Fremdsystemen von großer Bedeutung, da ein WfMS im Idealfall den Prozess „unsichtbar“ im Hintergrund steuert und dadurch ein hoher Integrationsgrad in bestehende Anwendungen notwendig ist. Es werden häufig folgende Schnittstellen von den Herstellern angeboten:

- **Daten-Schnittstelle**

Diese ermöglicht es, Daten aus anderen Anwendungen für ein WfMS zur Verfügung zu stellen, um sie für die Steuerung des Prozesses zu verwenden und die Bearbeitung der Daten durch den Sachbearbeiter zu ermöglichen.

- **Programm-Schnittstelle**  
Damit können Programme bzw. Funktionen aus anderen Programmen angesprochen werden, um aufgrund eines Prozesszustandes ein Programm oder eine Funktion aus einem Programm aufrufen zu können.
- **Benutzer-Schnittstelle**  
Diese entspricht einem Workflow-Client, der durch den WfMS-Anbieter als vorgegebene Komponente zur Verfügung gestellt wird oder durch den Kunden implementiert wird, um eine bessere Integration in vorhandene Lösungen zu ermöglichen.

### **Simulationskomponente**

Die Simulationskomponente hat sich bereits als Standardkomponente in den gängigen Modellierungswerkzeugen bzw. Modellierungskomponenten etabliert. Sie soll den Ablauf des modellierten Prozesses simulieren können, um Schwachstellen und Verbesserungsmöglichkeiten frühzeitig aufzeigen zu können. Für diesen Zweck müssen in der Modellierungskomponente die Bearbeitungswahrscheinlichkeiten und die angenommenen Zeiten (Bearbeitungs-, Liege- und Transportzeit) für die Aktivitäten eingegeben werden. Wurde dies für alle Aktivitäten spezifiziert, gibt man für die Simulation noch die Anzahl der Durchläufe bzw. die Anzahl der zu startenden Fälle für einen bestimmten Zeitbereich an und startet die Simulation. Anhand der grafischen Darstellung des Geschäftsprozesses kann dann festgestellt werden, in welchem Zweig des Prozesses es zu ungewollten Verzögerungen kommt und an welchen Stellen man den Prozess reorganisieren muss. Hierbei ist anzumerken, dass die Simulation nur so gut sein kann wie die zugrunde liegenden Modelldaten und daher auch nur das Modell prüfen kann. In der Praxis ergeben sich aber manchmal erhebliche Unterschiede zwischen dem Modell und den Abläufen im realen Produktionsprozess. Daher sollte die Simulationskomponente auch nach der Realisierung eines Geschäftsprozesses eingesetzt werden, um die tatsächlichen Engpässe der Lösung zu visualisieren. In diesem Fall werden die Laufzeitdaten aus der Workflow-Engine in die Simulationskomponente importiert, um sie dort auswerten zu lassen. Im Idealfall werden Engpässe grafisch dargestellt und entsprechende Verbesserungen vorgeschlagen.

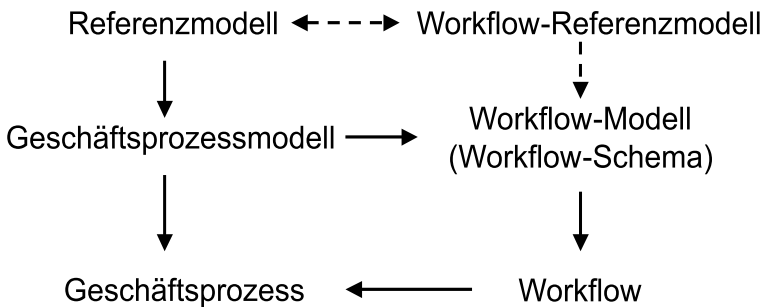
Die oben genannten WfMS-Komponenten kommen in unterschiedlichen Phasen zum Einsatz. Es wird hier klar zwischen einer Modellierungs-, Laufzeit- und Monitoringphase unterschieden. In der Modellierungsphase wird der Geschäftsprozess definiert und die Abarbeitung der Aktivitäten in einer chronologischen, regelbasierten Reihenfolge festgelegt. Zur Laufzeit werden primär die Geschäftsprozesse ausgeführt, die in der Monitoringphase überwacht und ausgewertet werden. Die aufgeführten WfMS-Komponenten und Funktionalitäten sind nicht zwingend in jedem Produkt realisiert. Darauf muss bei der Entscheidung für ein Produkt entsprechend der eigenen Anforderung geachtet werden, da in der Regel nur Teilbereiche eines vollständigen WfMS abgedeckt werden.



## 1.4 Workflow vs. Geschäftsprozess

Diese beiden Begriffe liegen sehr eng zusammen. Am einfachsten kann man die Begriffe durch die Sichtweise unterscheiden. Der Geschäftsprozess wird aus der Unternehmenssicht dargestellt. Er beschreibt die Kernprozesse und deren Hilfsprozesse der Wertschöpfungskette eines Unternehmens. Aus der technischen Sicht unterstützt der Workflow einen Geschäftsprozess komplett oder nur in Teilen durch die Verarbeitung der Informationen und durch eine Automatisierung des Geschäftsprozesses. Daraus ergeben sich unterschiedliche Ziele bei der Modellierung eines Geschäftsprozesses. Steht bei der Modellierung der Geschäftsprozesse die Wirtschaftlichkeit und die Dokumentation im Vordergrund, so ist es bei der Workflow-Modellierung die technische Realisierung der Zugriffe auf die benötigte Fachfunktionalität und auf die betroffenen Fachdaten sowie die technische Unterstützung des Ablaufes. Allerdings wird hier schon deutlich, dass sich aus einem Geschäftsprozessmodell nicht automatisch das zugehörige Workflowmodell ergibt.

In den folgenden Kapiteln Prozessmodellierung und Workflow-Projekte werden die beiden Sichten vereint, da diese eindeutige Trennung in der ganzheitlichen Betrachtung einer Geschäftsprozessrealisierung nicht angebracht ist.



**Abbildung 1.2.** Geschäftsprozess vs. Workflow

### 1.4.1 Historie der Workflow-Management-Systeme

Die ersten Arbeiten an Produkten, die einen Geschäftsprozess elektronisch unterstützen sollten, fanden Ende der 70er und Anfang der 80er Jahre statt. Es ging hauptsächlich um die Unterstützung von betrieblichen Abläufen und die Koordination von räumlich und zeitlich getrennten Arbeitsgruppen. Zurzeit werden auf dem Markt unzählige Produkte angeboten, die mit dem Begriff Workflow in Verbindung gebracht werden. Es ist dabei aber sehr genau zu unterscheiden, was diese Produkte tatsächlich in Bezug auf Workflow-Funktionalität implementiert haben. Dazu trägt die folgende Darstellung der Entwicklung der WfMS bei. Es geht darum, dem Leser die Kenntnisse zu vermitteln, aus einer Produktbeschreibung zu erkennen, ob es sich bei diesem Produkt um ein WfMS im Sinne der WfMC handelt, oder ob dieses Produkt lediglich die Möglichkeit bietet, den internen Ablauf

des Geschäftsprozesses zu unterstützen. Zunächst erfolgt kurz die Beschreibung von Systemen, die zwar in eine workflow-basierte Lösung integriert sein können, selbst aber kein WfMS darstellen.

### **Dokumenten-Management-System (DMS)**

Ein DMS befasst sich mit der Verwaltung eines elektronischen Dokumentes über dessen kompletten Lebenszyklus. Dies beinhaltet die Versionisierung der unterschiedlichen Textblöcke eines Dokumentes, die Verwaltung der Berechtigung für Zugriffe auf ein Dokument oder auch nur auf Teile eines Dokumentes und das Weiterleiten eines Dokumentes, falls es unterschiedlichen Genehmigungs- oder Freigabeverfahren unterliegt. Im Mittelpunkt dieses Systems stehen das Dokument und die Aktivitäten rund um ein Dokument. Es ist nicht vorgesehen, andere Applikationen, die mit der Bearbeitung eines Dokumentes zu tun haben, anzubinden.

### **E-Mail-System**

Das E-Mail-System gibt einem Anwender die Möglichkeit, Informationen individuell an andere Anwender weiterzuleiten, welche sich innerhalb oder außerhalb der Organisation befinden. Durch organisationsspezifische Regeln kann der Fluss der E-Mails so beeinflusst werden, dass nur ein beschränkter Anwenderkreis bestimmte Aktivitäten durchführen kann. Ein E-Mail-System verwendet eine E-Mail als Basis für die Steuerung der Informationen. Es hat den Vorteil, dass es ressourcensparend realisiert werden kann und je nach Realisierung auch einfach firmenübergreifend betrieben werden kann. E-Mail-Systeme haben für die Steuerung eines Geschäftsprozesses entscheidende Nachteile: Es ist schwer, den Ablauf/Verlauf und Verbleib der E-Mail nachzuvollziehen und es ist mit einfachen Mitteln möglich, den geplanten Ablauf oder den Inhalt der Information zu verändern, ohne dass es von dem Prozessbesitzer gewünscht ist. Weiterhin sind E-Mails nicht revisionssicher, und es gibt in E-Mail-Systemen permanente Probleme mit Viren und anderen zerstörerischen Programmen.

### **Groupware-Applikationen**

Die Groupware-Applikationen sind Software-Pakete, die für eine bessere Zusammenarbeit innerhalb einer Gruppe und zwischen Gruppen konzipiert wurden. Die Applikationen unterstützen die Arbeit in Form von Ad-hoc-Prozessen, gruppenspezifischen Bekanntmachungen und durch einen gemeinsamen Terminkalender. Der Fokus solcher Applikationen ist es, einen formalen und kontrollierbaren gemeinsamen Arbeitsbereich zur Verfügung zu stellen.

### **Transaktionsbasierende Applikationen**

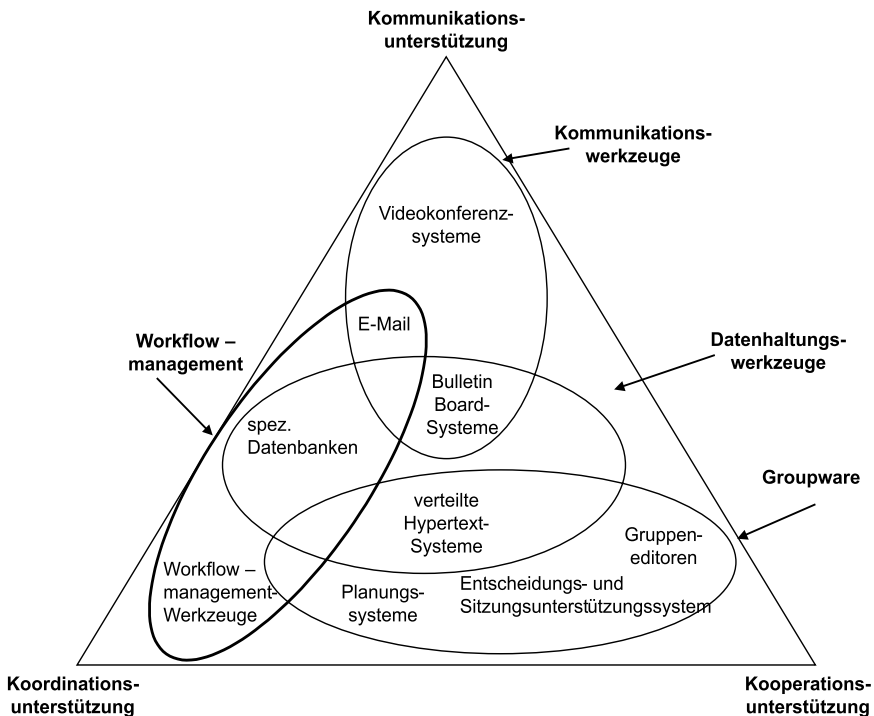
Transaktionsbasierende Applikationen wurden vor einigen Jahren entwickelt und unterstützen nur bestimmte Kategorien von Geschäftsverfahren. Die Anwendungen sind aufgrund der Geschäftsanforderungen für eine hohe Anzahl von Computerplattformen entwickelt worden und so zum damaligen Zeitpunkt auf einem relativ fortschrittlichen und richtungsweisenden technischen Standard gewesen. Transaktionsbasierende Applikationen stellen aufgrund ihrer Anforderungen keine Trennung zwischen der Geschäftslogik und den verschiedenen Anwendungen zur Verfügung. Diese Anforderung kam erst in den darauffolgenden Jahren auf.

## Projektmanagement-Applikationen

Projektmanagement-Applikationen wurden entwickelt, um komplexe Projekte in der Applikationsentwicklung zu unterstützen. Die Applikation enthält Workflow-Funktionalitäten, um einzelne Arbeitsschritte oder Informationen an die Projektbeteiligten gezielt weiterzuleiten und um den Projektverlauf zu kontrollieren. Als Folge dieser Funktionalitäten kann der aktuelle Projektstatus ohne zusätzlichen Aufwand an die Verantwortlichen kommuniziert werden. In einigen Produkten ist diese Art von Software generalisiert worden, um eine breitere, fachlich motivierte Ansicht des Prozesses und eine breitere Nutzungsmöglichkeit zu ermöglichen.

Der Markt für WfMS entwickelte sich somit über die Anforderungen der IT-Industrie und wurde in den unterschiedlichen Lösungen implementiert. Da diese Lösungen aber nur für einen sehr speziellen fachlichen Bereich oder für einen Teil der Geschäftsprozesse erstellt wurden, musste für die Verknüpfung der Teillösungen ein WfMS entwickelt werden, das es ermöglichte, alle Aktivitäten des Prozesses zu steuern und Prozesse auch unternehmensübergreifend zu unterstützen.

In Abbildung 1.3 wird eine Einordnung der aufgeführten Systeme nach Teufel (Teufel et al. 1995) dargestellt. An den Ecken werden die in einem Geschäftsprozess gewünschten Unterstützungsarten aufgeführt. Hierzu zählen die Kommunikations-, die Koordinations- und die Kooperationsunterstützung.



**Abbildung 1.3.** Einordnung von WfMS nach Teufel

## 1.5 WfMC (Workflow Management Coalition)

Die WfMC wurde primär gegründet, um ein Referenzmodell für ein WfMS zu entwickeln mit dem Ziel, herstellerunabhängig Module eines WfMS miteinander verknüpfen und betreiben zu können.

Die Vereinigung wurde 1993 von über 100 Organisationen (Entwickler und Anwender von WfMS) gegründet und besteht z.Zt. aus ca. 180 Herstellern von WfMS-Produkten, Anwendern und Beratungsunternehmen. Das Referenzmodell definiert eine Architektur mit den Hauptkomponenten und Standardschnittstellen, um zwischen den WfMS und den einzelnen Komponenten und Werkzeugen wie z.B. Modellierungstools zu kommunizieren. Die Vereinigung hat zum Ziel, dass Prozesse zwischen unterschiedlichen WfMS und den Modellierungswerkzeugen der unterschiedlichen Hersteller ausgetauscht und betrieben werden können. Somit soll der Aufwand und das Risiko für die Anbindung zwischen den WfMS und den Modellierungswerkzeugen so gering wie möglich gehalten werden.

Ein WfMS ist nach der WfMC in drei funktionale Bereiche aufgeteilt. Diese Bereiche orientieren sich nach dem Erstellen, Betreiben und Kontrollieren eines Prozesses. Man unterscheidet dabei zwischen der *Build-Time*-, der *Run-Time Process Control*- und der *Run-Time Activity Interaction-Funktionalität*.

Mit der *Build-Time-Funktionalität* werden die Prozesse aus der realen Welt in eine Computer-lesbare Definition übersetzt, nachdem der Prozess analysiert, optimiert und modelliert wurde. Für die Instanziierung und Steuerung der Prozesse steht die *Run-Time Process Control-Funktionalität* zur Verfügung, und für die Interaktion zwischen Anwender und computergestütztem Prozess wird die *Run-Time Activity Interaction* zur Verfügung gestellt (vgl. Abbildung 1.4). Das Workflow-Referenzmodell definiert diese Funktionalitäten und gibt einen sehr guten Überblick über das, was die WfMC anstrebt.

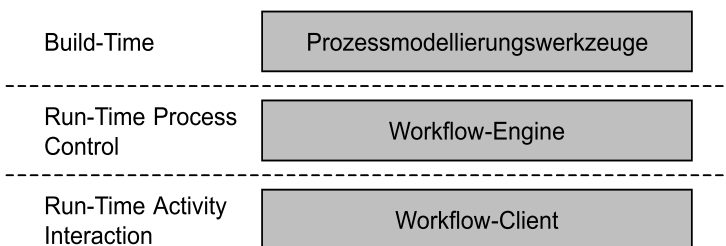


Abbildung 1.4. WfMS-Charakter

### 1.5.1 Workflow Reference Model

Im Workflow Reference Model wird der Aufbau eines WfMS definiert. Dieses Kapitel wird das Modell, seine Komponenten und die Schnittstellen beschreiben, um einen allgemeineren Überblick über ein WfMS zu erhalten. Zurzeit sind noch

nicht alle Schnittstellen verabschiedet und müssen daher mit der entsprechenden Vorsicht betrachtet werden; die meisten Schnittstellen befinden sich noch in der Entwicklung. Das Interface 4 (vgl. Abbildung 1.5) ist das einzige endgültig verabschiedete Interface. Der Definitionsvorschlag für das Interface 2 ist lediglich vorgestellt. Im Folgenden werden die sechs Komponenten mit den Interfaces und den Teilkomponenten beschrieben.

Die Laufzeitumgebung der Prozesse wurde von der WfMC *Workflow Enactment Services* genannt und beinhaltet mindestens eine Workflow-Engine, das Workflow Application Programming Interface (WAPI) und das Übergabeformat (Interchange Format) der Daten. Die Funktionen des WAPI wurden in den unterschiedlichen Interfaces gruppiert und getrennt betrachtet. Durch die Definition der WAPI kann gewährleistet werden, dass die Hersteller der unterschiedlichen WfMS und Tools die bearbeiteten Daten austauschen können. Durch die Gruppierung ist neben der fachlichen Trennung auch erreicht worden, dass die Interfaces getrennt betrachtet und verabschiedet werden können. Bevor die einzelnen Interfaces beschrieben werden, wird auf die Komponente eingegangen, die im Mittelpunkt des Modells steht. Der *Workflow Enactment Service* ist eine Laufzeit-Umgebung, die aus mindestens einer Workflow-Engine besteht und in der die Prozessinstanzen erstellt, verwaltet und ausgeführt werden. Der Service wird bzw. kann über das WAPI von „außen“ mit dem Interchange Format angesprochen werden und somit den Geschäftsprozess in der Workflow-Engine beeinflussen. Neben dem *Workflow Enactment Service* werden die folgenden Interfaces durch die WfMC definiert:

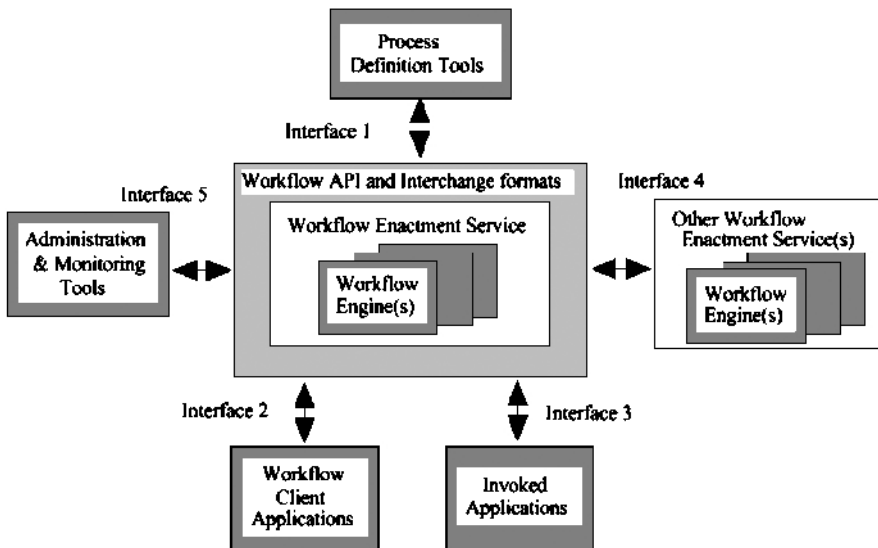


Abbildung 1.5. WfMC Workflow Reference Model

**Interface 1**

Das Interface 1 (Process Definition Interface) dient zur Anbindung des Modellierungswerkzeuges an die Workflow-Engine. Die Beschreibung definiert das Datenformat für den Austausch der Prozessdefinition und die Funktionsaufrufe in dem WAPI. Dadurch ist es möglich, dass ein Modellierungswerkzeug des Herstellers A der Workflow-Engine des Herstellers B die Daten für einen Prozess liefert. In dem Interface 1 wird die Übernahme von Aktivitäten eines Prozesses inklusive der verwendeten Applikationen, Daten und deren Datentypen beschrieben. Für den Prozess werden auch die Bedingungen für den Start, den Ablauf in Form von Zustandsübergangsbedingungen und für das Ende definiert. Zusätzlich zu den Bedingungen werden auch die Rollen übergeben, die den Aktivitäten zugeordnet sind. In der Definition ist neben den Daten auch die Struktur eines Prozesses festgelegt, welche sowohl im Modellierungswerkzeug als auch in der Workflow-Engine hinterlegt sein muss.

**Interface 2**

Das Interface 2 (Workflow Client Applications) definiert die Schnittstelle zwischen der Workflow-Engine und den Workflow-Clients. Der Client wird häufig mitgeliefert, um dem Akteur die Bearbeitung seiner Aktivitäten zu ermöglichen. Eine zweite Variante ist, anhand der Schnittstellendefinition einen eigenen Client zu entwickeln. Dieser Client enthält dann einen an das Unternehmen angepassten Funktionsumfang, um die Prozesse in der Workflow-Engine zu bearbeiten. Wurde der Client und die Workflow-Engine unter Einhaltung des Standards entwickelt, so ist es theoretisch möglich, dass der Workflow-Client von Produkt A die Prozesse der Workflow-Engine von Produkt B bearbeiten kann. Praktisch spielt dies jedoch keine Rolle, da die Workflow-Clients auf die jeweilige spezielle Funktionalität der Workflow-Engine abgestimmt sind.

**Interface 3**

Das Interface 3 (Invoked Applications) regelt die Anbindung von Software-Modulen, die für den Prozess erforderlich sind, wie z.B. einer Textverarbeitung, einem ERP-Produkt oder einem E-Mail-System. Die WfMC unterscheidet hier zwischen den invoked und den enabled Applications. Die invoked Applications werden über einen Agenten an die Workflow-Engine angebunden. Diese Applikationen können lokal auf dem Server der Workflow-Engine oder remote, also über das Netzwerk erreichbar, installiert sein. Eine weitere Möglichkeit ist die Anbindung von Applikationen anhand der enabled Application-Definition. In diesem Fall wird die Applikation direkt über eine Schnittstelle angebunden.

**Interface 4**

Das Interface 4 (Other Workflow Enactment Services) ist für die Anbindung weiterer WfMS definiert worden. Es werden für diese Anbindung zwei Arten berücksichtigt. Es kann zum einen eine Prozessdefinition weitergegeben werden, um z.B. einen Teil des Prozesses in einer Workflow-Engine zu starten und in einer anderen fortzuführen. Eine weitere Möglichkeit der Anbindung besteht darin, Prozessdefinitionen zwischen den Workflow-Engines auszutauschen. Es kann dann ein Prozess in der Workflow-Engine definiert und in einer anderen instanziiert werden.

### Interface 5

Mit der Definition des Interface 5 (Administration and Monitoring Tools) wird festgelegt, wie Administrations- und Überwachungswerkzeuge von Drittanbietern an die Workflow-Engine angebunden werden können. Mit diesen Werkzeugen werden die Auswertungen der bearbeiteten Prozesse erstellt, unabhängig davon, in welchem WfMS diese ausgeführt wurden. Der Anwender hat dank des Standards die Möglichkeit, herstellerunabhängig sein Werkzeug auszusuchen und kann somit ein Werkzeug wählen, welches für seine Anforderungen am besten geeignet ist.

Das Referenzmodell ist nur ein Teil der Standards, die von der WfMC angestrebt werden. Die Organisation setzt sich außerdem noch mit weiteren Standards auseinander, die in separaten Dokumenten veröffentlicht werden. Im Anhang befindet sich eine Tabelle, welche die Dokumente mit dem z.Zt. aktuellen Status (Stand Juli 2004) beinhaltet.

### 1.5.2 Metamodell eines Workflows

Die WfMC hat ein Metamodell entwickelt, welches die Grundstruktur eines Workflows definiert. Es zeigt die Zusammenhänge zwischen den Objekten und die Mindestanforderungen der Objektbeschreibungen des Prozesses. In Abbildung 1.6 und in der darauf folgenden Aufzählung kann man erkennen, dass das Meta-Modell an die Software-Architektur eines WfMS angelehnt ist. Die Attribute beschreiben, welche Daten zwischen den WfMS-Komponenten ausgetauscht werden.

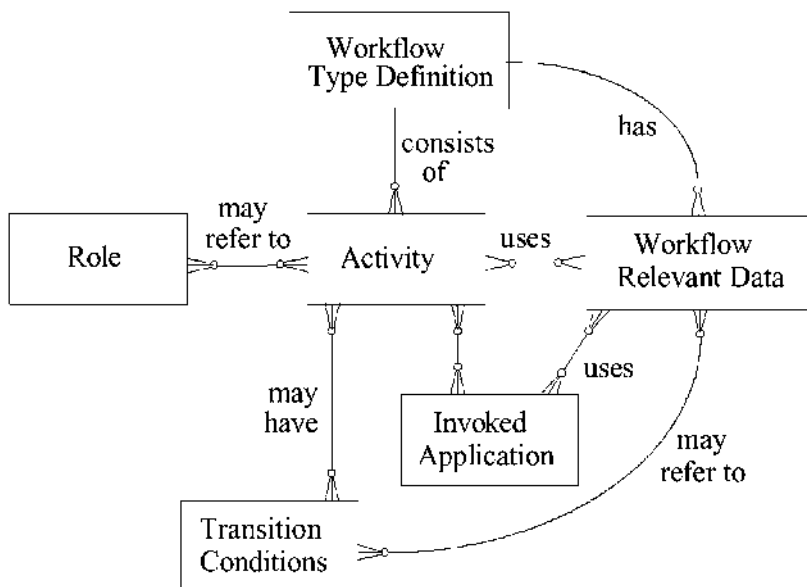


Abbildung 1.6. WfMC Process Definition Meta Model

Die aufgeführten Attribute sind als Mindestanforderungen an einen Prozess zu verstehen und können durch die Produkthersteller erweitert werden.

**Workflow-Typdefinition (Workflow Type Definition)**

- Prozessname
- Versionsnummer
- Start- und Endebedingung des Prozesses
- Sicherheits-, Prüf- oder andere Kontrolldaten

**Aktivität (Activity)**

- Name der Aktivität
- Typ der Aktivität
- Bedingungen für die Pre- und Post-Aktivitäten
- Andere zeitliche Beschränkungen

**Übergabebedingungen (Transition Conditions)**

- Fluss- oder Ausführungsbedingungen

**Workflow-bezogene Daten (Workflow Relevant Data)**

- Dateiname und Pfad
- Datentyp

**Rolle (Role)**

- Name und organisatorische Verknüpfungen

**aufzurufende Applikation (Invoked Application)**

- Typ und Name der Applikation
- Benötigte Parameter für die Ausführung der Applikation
- Pfad zur Applikation

## 1.6 Softwarearchitektur einer workflow-basierten Lösung

Anhand der Architektur erkennt man deutlich die Unterschiede zwischen einer Applikation wie z.B. einer Groupware (E-Mail)-Lösung und einer Lösung, welche auf einem WfMS basiert. Ein WfMS ist meistens eine datenbankbasierte Client/-Server-Lösung und kann die Vorteile einer relationalen Datenbank ausnutzen. Die größten Vorteile sind hier das Transaktionshandling und die Datensicherheit der Datenzugriffe und Datenverwaltung. Werden die Datenbanken der Marktführer in der Architektur des WfMS verwendet, so kann zusätzlich davon ausgegangen werden, dass große Datenmengen, also eine hohe Anzahl von gestarteten Geschäftsprozessen und deren Daten, durch die Datenbank bearbeitet werden können, ohne Leistungseinbußen zu bemerken. Dies ist u.a. darin begründet, dass Datenbanken besser als E-Mail-basierte-Systeme skaliert werden können. Im Folgenden wird die logische Struktur einer Lösung beschrieben, die ein WfMS ent-



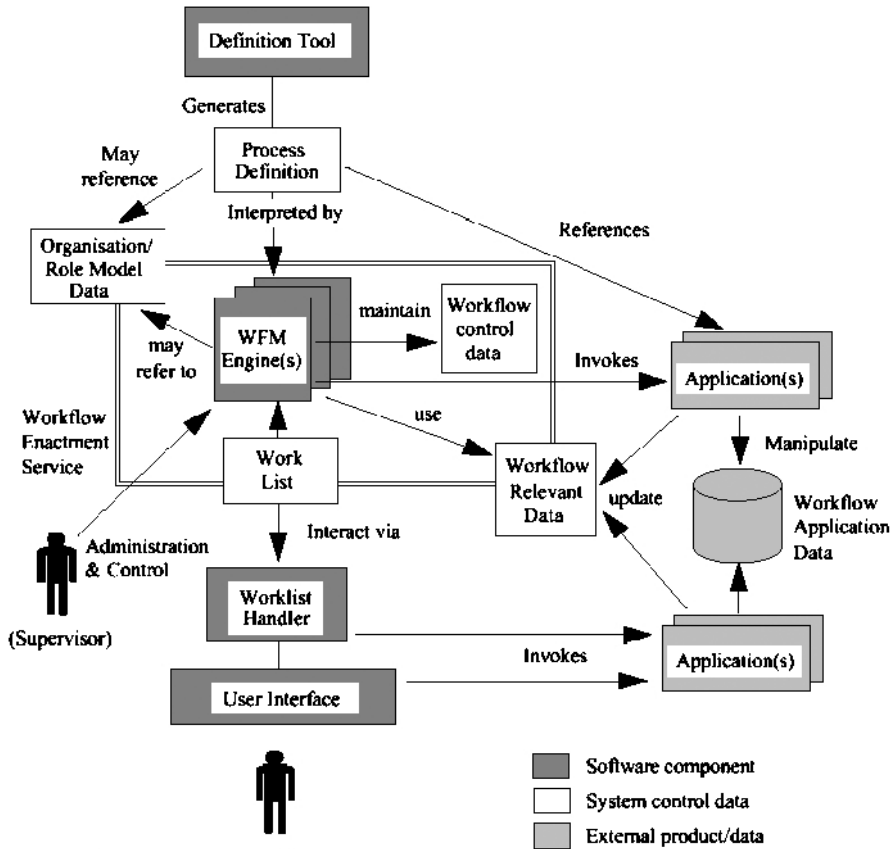


Abbildung 1.7. WfMC Workflow-Produktstruktur

hält. Es wird gezeigt, wie die einzelnen bereits beschriebenen Komponenten miteinander agieren und in welchen Phasen die Komponenten zum Einsatz kommen.

Die Abbildung 1.7 entspricht der Abbildung *Generic Workflow Product Structure* aus dem Workflow-Referenzmodell (TC00-1003) der WfMC. Zusätzlich zu den vorher beschriebenen werden weitere Komponenten und deren Beziehungen untereinander aufgezeigt. Zwischen der Modellierungskomponente und dem Organisations- und Rollenmodell besteht eine Verbindung, um den zu modellierenden Prozessen die Rollen und damit die Akteure zuordnen zu können. Wird in einem Unternehmen angestrebt, das Rollenmodell außerhalb des WfMS zu halten und zu pflegen, so ist darauf zu achten, dass beide Produkte eine abgestimmte Schnittstelle für den Datenaustausch zur Verfügung stellen. In wenigen WfMS-Produkten wird dies durch eine LDAP-Schnittstelle (Lightweight Directory Access Protocol) zwischen den beiden Modulen ermöglicht.

Ein weiterer Aspekt ist die Anbindung der Fachanwendungen. Die Anbindung an das WfMS kann auf mehrere Arten geschehen. Da während der Modellierung die Fachanwendungen, die in den Prozess mit eingebunden werden sollen, bekannt sein müssen, besteht eine Schnittstelle zwischen der Modellierungskomponente und den Fachanwendungen. Dies muss nicht die Fachanwendung als Ganzes sein, sondern kann die durch den Anwendungshersteller dokumentierte Funktionalität der Anwendung betreffen.

Sollen Fachanwendungen eingebunden werden, müssen diese eine Schnittstelle vorweisen können, um die Funktionalitäten von „außen“ ansprechen zu können. Eine Anbindungsart wird nach der Modellierungsphase der Geschäftsprozesse bestimmt. Während der Realisierungsphase des Geschäftsprozesses müssen die Entwickler die für den Prozess verwendeten Funktionen der Fachanwendungen direkt an die Workflow-Engine anbinden. Dies wird programmtechnisch über die veröffentlichten APIs (Application Programming Interfaces) der Fachanwendungen realisiert. Die APIs müssen die Funktionen enthalten, um die Aktivität bearbeiten zu können, die ihr zugeordnet wurde. Eine weitere Anbindungsart an die Fachanwendungen wird im Gegensatz zu den anderen Arten direkt auf dem Client-PC realisiert. Die lokal auf dem Client installierten Fachanwendungen werden direkt von dem Workflow-Client angesprochen. Dies hat zur Folge, dass die Workflow-Engine nicht direkt mit den lokalen Fachanwendungen kommunizieren kann, sondern nur die prozessrelevanten Daten, die im WfMS gepflegt werden und die von der Fachanwendung bearbeitet werden müssen, austauscht. Beispielsweise kann so eine Textverarbeitung wie Microsoft Word an den Workflow-Client an einem Arbeitsplatz angebunden werden. Neben der eingeschränkten Kommunikation zwischen der Anwendung und der Workflow-Engine kommt hinzu, dass dieser PC-Arbeitsplatz einen höheren Aufwand bei der Software-Verteilung erfordert, da zu einem Workflow-Client immer Word mit installiert sein muss.

Der Workflow-Client dient als zentrale Arbeitsplattform für den Sachbearbeiter. Die Workflow-Engine stellt dem Sachbearbeiter (Akteur) die Aktivitäten gemäß seiner Rollenzuordnungen bereit. Somit erhält er über seine Aktivitätenliste die Aktivitäten, die er bearbeiten kann. Die Darstellung der Aktivitäten kann, bezogen auf die unterschiedlichen Prozesse, von Client zu Client unterschiedlich sein. Aktivitäten des Prozesses, für die der Akteur keine Rollenzuordnung hat, kann er auch nicht einsehen. Einige Workflow-Clients stellen für den bisherigen Verlauf ein Historien- bzw. ein Ablaufdiagramm zur Verfügung, um einen Einblick in den bereits erfolgten Ablauf zu erhalten. Der Sachbearbeiter hat über die Aktivitätenliste keinen Überblick darüber, welche Fachanwendungen gerade für die Aktivität verwendet werden. Dies ist auch nicht erforderlich, da primär die Durchführung einer Aktivität im Vordergrund steht. Dazu kommt, dass durch die „Maskierung“ der Fachanwendungen diese meistens einfach austauschbar werden, ohne die Anwender neu schulen zu müssen.

An dieser Stelle soll auch der Unterschied zwischen den Prozesskontrolldaten und den prozessrelevanten Daten erwähnt werden. Die Prozesskontrolldaten umfassen alle Steuerungsdaten, die für den Ablauf des Prozesses benötigt werden. Dazu ge-

hören die zugeordneten Aktivitäten und Bedingungen für den Ablauf des Geschäftsprozesses sowie die Informationen, welchen Akteuren diese Aktivitäten bereitgestellt werden sollen, aber auch der aktuelle Zustand und die Historie des Prozesses. Die prozessrelevanten Daten sind die fachlichen Daten, die für diese Prozessinstanz und damit für die Bearbeitung des eigentlichen Geschäftsprozesses notwendig sind, wie z.B. die Adresse des Antragstellers und die Höhe der Kreditsumme.

In dieser Architekturbeschreibung wird nicht auf die Technologie eingegangen, mit der die einzelnen Produkthersteller ein WfMS realisiert haben. Da diese nicht standardisiert wurde, sind die Produkte und deren WfMS-Komponenten durch unterschiedliche Technologien realisiert worden. Mögliche technische Architekturen, die in einer WfMS-Realisierung zur Anwendung kommen können, werden im Kapitel Software-Architekturen erklärt.

## **1.7 Warum Workflow einführen?**

Der zunehmende Konkurrenzdruck und die Globalisierung fordern heute eine konsequente Ausrichtung der Unternehmen am Kunden. Dabei muss der Service bei gleichzeitig effizienter Nutzung der zur Verfügung stehenden Ressourcen in den Vordergrund rücken. Die Themen Prozessoptimierung und Kostensenkung gewinnen eine immer höhere Bedeutung. Aber auch Kundenbindung und Kundenservice ist eine wesentliche Anforderung an die heutige Geschäftswelt. Nur höchste Standards bezüglich Produktqualität, Effizienz und vor allem Qualität des Service können die Existenz und Wettbewerbsfähigkeit heutiger Unternehmen bei zunehmender Globalisierung und immer ähnlicheren Produkten sichern. Zur Schaffung dieser Grundlagen gilt es, die Geschäftsprozesse im Unternehmen zu analysieren, zu dokumentieren, zu optimieren und dort, wo es sinnvoll ist, zu automatisieren. Dies führt zur Verkürzung der Bearbeitungszeiten, Schaffung besserer Informationsgrundlagen und Erhöhung der Effizienz und Wirtschaftlichkeit des Unternehmens – die Voraussetzungen, um dem Kunden einen optimalen und zufriedenstellenden Service zu bieten und ihn dauerhaft an das Unternehmen zu binden. Workflow-basierte Lösungen ermöglichen es, die arbeitsteiligen Geschäftsprozesse des Unternehmens zu automatisieren, aktiv zu steuern und damit zu optimieren und effizient zu gestalten. Ein WfMS begleitet den ständigen Prozess der Reorganisation und versetzt die Unternehmen in die Lage jederzeit rasch auf sich ändernde Marktanforderungen zu reagieren. Weiterhin deckt eine workflow-basierte Lösung viele Anforderungen an ISO 9000ff-zertifizierte Arbeitsverfahren ab.

### **1.7.1 Aktuelle Situation in den Unternehmen**

Die eingesetzte Software und die große Anzahl vernetzter Beziehungen in Unternehmen sind Ursache der zunehmenden Komplexität von Geschäftsvorfällen und ihrer Bearbeitung. In der Folge ist die Bearbeitung selbst für den einzelnen Mitarbeiter häufig unüberschaubar und nicht mehr im Detail nachvollziehbar. Bei stark

strukturierten Geschäftsprozessen verteilen sich über 75% der Dauer eines Vorgangs auf Informationsbeschaffung der Mitarbeiter sowie Wege- und Liegezeiten. Oft wird ein sehr geringer Teil der Dauer des Geschäftsprozesses (Bearbeitungszeit) zur produktiven Bearbeitung der eigentlichen wertschöpfenden Aktivitäten verwendet. In einer Studie<sup>2</sup> wird dieser Anteil mit weniger als 15% angegeben. Insgesamt herrschen heute in Unternehmen ohne integrierende workflow-basierte Lösungen häufig folgende Defizite:

- lange Durchlaufzeiten der Geschäftsprozesse
- Vergessen von Aktivitäten
- Verstreichen von Fristen
- fehlende notwendige Informationen
- keine Nachvollziehbarkeit des Geschäftsprozesses
- ineffizientes Arbeiten der Sachbearbeiter
- Probleme bei Ausfall von Mitarbeitern
- unsichere Bearbeitung der Sachbearbeiter
- auf Kapazitätsengpässe kann nicht zeitnah reagiert werden
- lange Einarbeitungszeit der Mitarbeiter und
- Fehleranfälligkeit durch eine Anwendungsvielfalt

### **1.7.2 Vorteile von workflow-basierten Lösungen**

In der Summe führen die oben aufgeführten Punkte zu lang andauernden und dadurch teuren Bearbeitungen, unzufriedenen Kunden, aber auch zu unzufriedenen und verunsicherten Mitarbeitern. Genau in diesen Bereichen erwartet man sich von workflow-basierten Lösungen erhebliche Verbesserungen. In der oben genannten Studie werden von den befragten Unternehmen folgende Fachziele bei der Einführung von Workflow genannt:

- Beschleunigung der Durchlaufzeiten
- Kanalisierung der Informationsflut
- Verbesserung der Kommunikation
- Transparenz komplexer Vorgänge
- Ablauf-/aufgabenorientierte Systemunterstützung
- präzise Führung in der Sachbearbeitung
- Integration von Anwendungsiseln
- Dokumente gezielt verfügbar machen
- präzises Prozessmanagement
- Verbesserung der Umsetzung von Prozessoptimierung
- Reduzierung von Individualprogrammierung und
- Prozessoptimierung durch Workflow-Daten

---

<sup>2</sup> WORKFLOW-TRENDS 2000: Computerwoche Studie, Unterwössen: Peschanel & Partner GmbH, 1999

Als Unternehmensziele tauchen in der Studie auf:

- Effizienzsteigerung
- Steigerung der Bearbeitungsqualität
- höhere Wirtschaftlichkeit
- Kostensenkung
- bessere Nutzung von Netzinfrastrukturen

Von relativ geringer Bedeutung ist der Wunsch nach Personaleinsparungen. Abhängig vom Unternehmen ergeben sich beim Einsatz workflow-basierter Lösungen noch die folgenden Vorteile:

- klare Zuständigkeiten
- automatische Verteilung der Arbeitslast
- Fristenkontrolle und Eskalation
- aktuelle Informationen zum Bearbeitungszustand
- Vermeidung von Medienbrüchen und
- Grundlagen für ISO 9000-Zertifizierung

Das führt bei richtig eingesetzten workflow-basierten Lösungen für das Unternehmen zu den folgenden Nutzenaspekten:

- Zeit- und Kostenersparnis
- Termintreue
- bessere Kundenbindung (z.B. durch schnelle Auskünfte bei Kundenanfragen)
- Fehlerreduzierung
- effektives Controlling
- erhöhte Mitarbeiterzufriedenheit und
- Sicherstellung von Qualitätsstandards gemäß ISO 9000ff

Neben den dargelegten Vorteilen hat ein Projekt zur Einführung von Workflow-Management auch erhebliche Herausforderungen:

- Auf ein Unternehmen kommen zunächst zusätzlich Kosten für Software und eventuell auch für Hardware zu. Auch müssen die Kosten für die Wartung bzw. die Reorganisation der Prozesse eingeplant werden.
- Die Mitarbeiter müssen auf die eingeführte Software und neue Arbeitsweisen geschult werden.
- Der Betriebsrat ist frühzeitig in das Projekt mit einzubeziehen.
- Wird nicht der komplette Ablauf der Prozesse durch ein Workflow-Management abgedeckt, muss dann mit Medienbrüchen gerechnet werden, die zusätzlichen Aufwand in der Handhabung bedeuten können. Andererseits ist das meistens immer noch besser als die Ausgangssituation.

- Zum Teil entsteht ein erheblicher Aufwand bei der Integration von Anwendungen. Es muss darauf geachtet werden, in welcher Form eine Anwendung mit dem WfMS verbunden wird. Altanwendungen wie z.B. Hostanwendungen können durch Drittanbieter angebunden werden. Dies bedeutet einen weiteren Kostenaufwand für Lizenzen.

Nachdem die Vorteile und Herausforderungen einer Workflow-Einführung dargelegt wurden, stellt sich nun die Frage: Woran erkennt man, welche Geschäftsprozesse sich für eine Workflow-Unterstützung eignen? Die offensichtlichsten Kriterien sind die Häufigkeit und die Regelmäßigkeit eines Prozesses. Je größer die Anzahl der gestarteten Vorgänge und je höher der Grad eines standardisierten Ablaufes ist, um so besser eignet sich der Prozess für die Unterstützung durch ein WfMS. Auch die Anzahl der involvierten Mitarbeiter ist relevant. Kann durch eine Teilautomatisierung die Anzahl bzw. die Arbeitsbelastung verringert oder die Durchlaufzeit verbessert werden? Kann an der Anzahl der Beteiligten nichts verändert werden, so erreicht man durch die Unterstützung eines Workflows auch einen verbesserten Ablauf des Geschäftsprozesses, da das WfMS den Ablauf verwaltet und durch eine geschickte Modellierung die erforderliche Information immer mitführt. So können auch die Transportzeiten und Liegezeiten optimiert werden. Neben diesen quantitativen Kriterien ist auch die Struktur des Geschäftsprozesses ein wichtiges Merkmal für eine Workflow-Unterstützung. Ist der Geschäftsprozess leicht strukturiert, sind wenige Abteilungen und Schnittstellen zu Fachanwendungen mit dem Ablauf konfrontiert und der Ablauf seriell, so ist es relativ einfach, durch einen Workflow eine positive Veränderung zu erreichen, da dies leicht zu modellieren ist. Ein weiteres wichtiges Merkmal ist die Anzahl der Mitarbeiter pro Aktivität. Kann die Aktivität durch einen einzelnen Akteur durchgeführt werden, so muss dem Akteur diese Aktivität gezielt zugewiesen werden. Ein weiterer Auslöser für die Einführung von workflow-basierten Lösungen kann eine bevorstehende Häufung von Geschäftsprozessen sein, z.B. der Austausch von Kreditkarten oder Werbeaktionen. Anstatt neue Mitarbeiter einzustellen, führt man eine workflow-basierte Lösung ein, um den erhöhten Arbeitsanfall mit der gleichen Mitarbeiterzahl bewältigen zu können. Andere Kriterien sind beispielsweise sehr hohe „Strafkosten“ bei Terminverzögerungen oder die Anforderung nach Revisionssicherheit bearbeiteter Geschäftsvorfälle.

### 1.7.3 Prozesseigenschaften für den Einsatz eines WfMS

Bevor also ein Unternehmen ein WfMS einführt, ist zu prüfen, inwieweit eine Unterstützung der Prozesse durch ein solches System sinnvoll und möglich ist. Liegt für diesen Zweck keine Prozessdokumentation im Unternehmen vor, in der der Ablauf und die Prozesseigenschaften beschrieben sind, so muss diese durch eine Analyse der Geschäftsprozesse erstellt werden. Es ist darauf zu achten, dass solche Dokumentationen alle benötigten Fachfunktionen dokumentieren, um beurteilen zu können, ob ein Geschäftsprozess komplett oder nur in Teilen automatisiert werden kann. Im Folgenden werden die wichtigsten Prozesseigenschaften aufgeführt, die den Einsatz eines WfMS begründen können:

- Der Geschäftsprozess weist eine Struktur auf, die sich in derselben Form häufig wiederholt. Schlecht strukturierte Prozesse können nur sehr abstrakt abgebildet werden und können so nicht optimal unterstützt werden.
- Die Anzahl der Wiederholungen des Geschäftsprozesses ist hoch. Durch ein WfMS behält man die Kontrolle über den Geschäftsprozess und kann eine auslastungsgesteuerte Arbeitsverteilung durchführen.
- Die Automatisierung des Prozesses verkürzt die Transport- und Wartezeiten. Der Prozess wird dadurch kostengünstiger bearbeitet.
- Es wird ein Prozess-Controlling benötigt. Neben den Prozesszeiten werden auch Informationen über den Status und die Ressourcenauslastung benötigt.
- Für den Geschäftsprozess sind aufbauorganisatorische Regeln wie Vier-Augen-Prinzip, Stellvertreterregelungen und relative Adressierung (über Rollen) von Mitarbeitern notwendig.
- Das WfMS kann die Vielzahl von unterschiedlichen Datenverarbeitungs-Werkzeugen durch eine einheitliche Benutzeroberfläche (GUI) vereinen. Es werden dadurch die Einarbeitungs- und Bearbeitungszeiten eines Geschäftsprozesses optimiert.

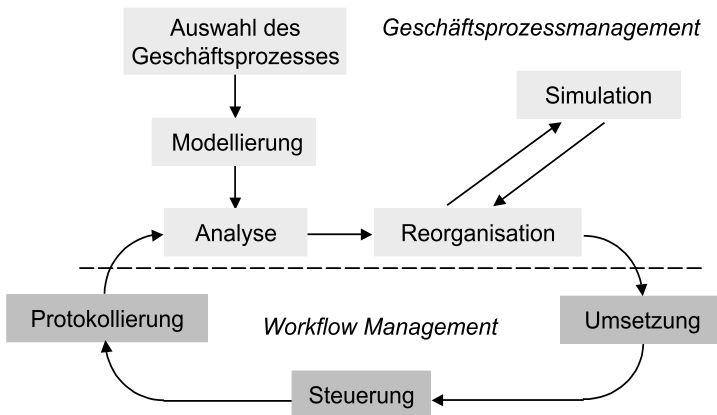
Ein weiteres Argument kann eine unternehmensübergreifende Bearbeitung eines Geschäftsprozesses sein. Optimal ist hierfür, wenn alle beteiligten Unternehmen den Einsatz eines WfMS unterstützen.

## 1.8 Der Workflow-Regelkreis

Intention bei der Einführung workflow-basierter Lösungen ist es, einen Regelkreis zwischen Modellierung und Prozesssteuerung aufzubauen, so dass die im Workflowsystem gewonnenen Laufzeitdaten in einem Modellierungswerkzeug ausgewertet und der Geschäftsprozess mit den gewonnenen Aufzeichnungen reorganisiert und optimiert werden kann. Führt man in einem Unternehmen zum ersten Mal eine Geschäftsprozessmodellierung durch, ist man häufig auf Vermutungen, Schätzungen oder recht ungenaue bzw. sehr aufwändige Ermittlungen der Zeiten (Bearbeitungs-, Transport- und Liegezeit) angewiesen. In den Simulationskomponenten der Modellierungswerkzeuge kann man die getroffenen Annahmen überprüfen. Ob diese aber mit der Realität übereinstimmen, zeigt sich erst im Produktiveinsatz des eingeführten WfMS. Daher ist es wichtig, die aus der Praxis gewonnenen Daten wieder dem Modellierungswerkzeug zuzuführen, dort zu optimieren und den geänderten Prozess wieder an das WfMS zu übergeben. Dieses Verfahren verdeutlicht die Forderung, dass die Software-Komponenten der Workflow-Lösung modular aufgebaut sein müssen. Durch einen aktivitätsorientierten, modularen Aufbau wird erreicht, dass ohne Programmieraufwand ein Prozess per Definition einen anderen Ablauf erhalten und so effektiv reorganisiert werden kann.

Abbildung 1.8 verdeutlicht, dass der Workflow-Regelkreis in zwei Kategorien unterteilt werden kann, die die oben genannten Bestandteile beinhalten. Die erste

Kategorie (Geschäftsprozessmanagement) setzt sich mit der Prozessdarstellung, -analyse und -optimierung auseinander. Dies kann durch Werkzeuge wie ARIS, Aeneis oder ADONIS unterstützt werden. Die zweite Kategorie (Workflow Management) setzt sich mit der Umsetzung und der Steuerung der erstellten Modelle auseinander. Als unterstützende Werkzeuge können hier allgemein die WfMS genannt werden. Durch die im WfMS protokollierten Prozesse können genaue Aussagen über die Prozesszeiten getroffen werden. Zusätzlich kann eine Auswertung der Daten in einem Modellierungswerkzeug die daraus folgenden Prozesskosten ermitteln. Es soll noch darauf hingewiesen werden, dass im Kapitel Prozessmodellierung die Unterscheidung zwischen den beiden Begriffen Geschäftsprozessmanagement und Workflow Management detaillierter erläutert wird.



**Abbildung 1.8.** Workflow-Regelkreis

Unabhängig vom Regelkreis gelten folgende Forderungen bezüglich der Modularität von Fachanwendungen:

1. Der Aufwand für eine Änderung des Prozessablaufes mit Hilfe der Modellierungswerkzeuge ist gering.  
Forderung: Die (ablauf-)organisatorischen Modifikationen dürfen nicht dazu führen, dass Änderungen in den Softwarekomponenten notwendig werden (modularer Aufbau).
2. Wenn die Analyse ergibt, dass bestimmte Tätigkeiten viel Zeit bei der Bearbeitung des Geschäftsprozesses benötigen, kann man gezielt an die Optimierung dieser Aktivitäten gehen.  
Forderung: Bei anderen Tätigkeiten im Geschäftsprozess dürfen prinzipiell keine Änderungen notwendig werden.
3. Wenn einzelne Softwarekomponenten innerhalb eines Unternehmens abgelöst bzw. umgestellt werden (z.B. Umstellung von Großrechner auf Client/Server), so dürfen von der Umstellung auch nur die jeweiligen Aktivitäten betroffen sein.  
Forderung: Bei anderen Tätigkeiten im Geschäftsprozess dürfen prinzipiell keine Änderungen notwendig werden.



Eine detailliertere und umfangreichere Beschreibung des Vorgehens zur Einführung eines WfMS wird im Kapitel Workflow-Projekte in Form eines Vorgehensmodells beschrieben.

## 1.9 Kommentar

Die hier vorgestellten Definitionen sind aufgrund der WfMC-Veröffentlichungen und eigener Projekterfahrungen entstanden. Plant ein Unternehmen die Einführung eines WfMS, sollte eine solche Definition der Begriffe im Vorfeld durchgeführt werden. Hierbei sind auch die weiteren Kapitel zu berücksichtigen. Es muss z.B. darauf geachtet werden, dass man sich bei der Modellierung eines Geschäftsprozesses immer auf derselben Abstraktionsebene (vgl. Kapitel 4. Prozessmodellierung) bewegt. Da jede dieser Abstraktionsebenen eigene Zielsetzungen beinhaltet, ist eine Vermischung der Betrachtungsweisen unbedingt zu vermeiden. Durch eine Trennung wird eine klare Darstellung der Ergebnisse erreicht. Eine einheitliche Definition im Unternehmen ist für eine effektive Einführung zwingend notwendig.

Um das Potential eines WfMS voll ausnutzen zu können, müssen zum Teil technische Herausforderungen bewältigt werden. So werden die Anbindungen an weitere IT-Systeme häufig unterschätzt. Es kommt bei der Anbindung darauf an, in welcher Art das IT-System angebunden werden soll. Es gibt auch hier mehrere Möglichkeiten (Daten- oder Funktionsanbindung), die im Kapitel Software-Architekturen noch genauer beschrieben werden. Hierbei muss während der Konzeption berücksichtigt werden, welche Anbindungsart den Aufwand für die Anbindung noch rechtfertigt. So kann im ersten Schritt die Anbindung erst mal so realisiert werden, dass der Anwender nur eine „Check“-Aktivität erhält, die ihn auffordert, in einem IT-System eine Aktion durchzuführen und dies durch einen Klick und ggf. optionalen Text dem WfMS zu bestätigen. Durch eine mögliche Erweiterung kann in einem zweiten Schritt das IT-System technologisch angebunden werden, falls dies zu einem besseren Kosten-Nutzen-Verhältnis führt.

Die Einführung eines WfMS birgt nicht nur technische Herausforderungen, sondern auch sehr hohe Anforderungen an die Akzeptanz der Mitarbeiter des Unternehmens. Je nach Branche und Kenntnisstand der Mitarbeiter kann eine Unterstützung der Geschäftsprozesse durch ein WfMS auch einen negativen Eindruck hinterlassen. Nicht selten werden Ängste um den Arbeitsplatz dadurch entfacht, dass sich die Arbeit nach einer WfMS-Einführung in der Regel effektiver darstellt und somit die gleiche Arbeit mit weniger Mitarbeitern durchgeführt werden kann. Außerdem kann durch die Monitoringfunktionalität bei dem Mitarbeiter der Eindruck entstehen, dass diese gewonnenen „Arbeitsdaten“ negativ gegen ihn ausgelegt werden können (Stichwort „Überwachung“). Dies ist prinzipiell möglich, aber die Aussagen über diese aufgezeichneten Daten müssen differenziert betrachtet werden. So erhält z.B. ein guter Mitarbeiter in der Regel die schwierigen Fälle und benötigt somit länger für die Bearbeitung als ein nicht so routinierter oder neuer Mitarbeiter, der die Standardfälle bearbeitet. Würde man nur anhand dieser Daten

einen Mitarbeiter beurteilen oder sogar entlassen wollen, wären relativ schnell die guten Mitarbeiter entlassen. Dieses Beispiel soll aufzeigen, dass diese Daten nur bedingt für die Beurteilung eines einzelnen Mitarbeiters zu gebrauchen sind.

Auch ist in großen Unternehmen frühzeitig der Betriebsrat in das Projekt und insbesondere bei der Gestaltung der Historie der Fälle und der Reports mit einzubeziehen. Es kann durch kleinere Veränderungen in der Darstellung der Daten eine positivere Wirkung erzielt werden.

Hierzu ein Beispiel aus eigener Erfahrung: Durch Ausblenden der Minutenangabe bei der Anzeige auf dem Bildschirm war die tatsächlich benötigte Bearbeitungszeit nicht sichtbar. Das WfMS kann aber intern die genaue Bearbeitungszeit verwenden und die Auswertungen über die gestarteten Fälle durchführen.

Weitere Akzeptanzprobleme können durch den veränderten und „erzwungenen“ Arbeitsstil entstehen. Der Mitarbeiter verliert die Freiheit aufgrund der technischen Unterstützung. Außerdem muss er sich von seinem persönlichen Arbeitsstil und der damit verbundenen informellen Kommunikation lösen.

### 1.9.1 Prozessdefinition

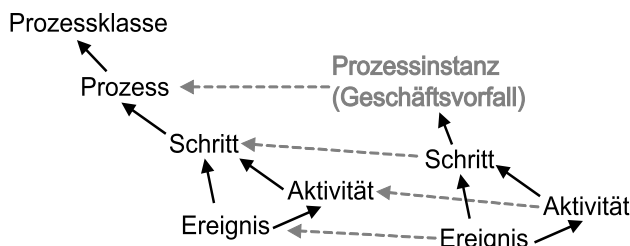
Die Prozessdefinition der WfMC sieht einen Prozess und die ihm zugeordneten Aktivitäten vor. Die zugeordneten Aktivitäten stehen in einer 1-zu-n-Beziehung und werden von Ereignissen gestartet und beendet. Wurde ein Prozess gestartet, so wird eine Prozessinstanz erzeugt, welche sich auf einen bestimmten Geschäftsvorfall bezieht. Zusätzlich zu dieser Prozessdefinition will der Autor noch auf eine weitere Definition aufmerksam machen, die sich in Projekten zur Einführung von workflow-basierten Lösungen bewährt hat. Dabei werden die unterschiedlichen Entwicklungsphasen eines Prozesses berücksichtigt. In der ersten Phase wird ein Prozess in einer „Entwicklungsumgebung“ definiert. Diese muss den vollen Funktionsumfang einer Produktionsumgebung vorweisen bzw. diese simulieren können.

Vor dem eigentlichen Beginn der Prozessdefinition wird eine Prozess-Klasse angelegt, in der alle artverwandten Prozesse zusammengefasst werden. Somit erhält man eine Gruppierung der Prozesse wie z.B. Kreditantrag, in der alle Arten von Kreditvergabeprozessen zusammengefasst werden. In der Prozess-Klasse sind alle Prozesse definiert, aus denen in der IT-Produktionsumgebung eine Prozessinstanz erstellt werden kann. Ein solcher Prozess enthält wiederum beliebig viele Schritte, denen alle Aktivitäten zugeordnet werden, die durch einen Akteur zum Zeitpunkt der Bearbeitung (an einem Arbeitsplatz) für die Bearbeitung des Geschäftsvorfalles benötigt werden.

Die Schritte können wie die Aktivitäten Ereignisse empfangen und Ereignisse auslösen. Diese Ereignisse werden durch die Workflow-Engine verwaltet. Den Schritten sind während der Definition die Ereignisse als Eingangs-(Input-) oder Ausgangs-(Output-) Ereignisse zugeordnet worden. Die definierten Ereignisse können

einzelnen oder als Kombination den Schritten zugeordnet werden. Diese Art der Struktur enthält zusätzlich zu der Definition der WfMC eine logische Gruppierung der Aktivitäten durch Schritte. Dies hat den Vorteil, dass immer wiederkehrende Schritte in einem Pool für zukünftig zu definierende Prozesse zur Verfügung gestellt werden können und somit der Aufwand für die Definition weiterer Geschäftsprozesse gering gehalten werden kann. So können auch Standardschritte erstellt werden, die als Schablone zu verwenden sind.

Die Vorteile dieser Definition sind sehr leicht zu erkennen. Mit einfachen Mitteln kann so die Effizienz und die Qualität der Definition gesteigert werden. Zudem ist in einem Unternehmen die Rolle eines Prozessadministrators etabliert, der die Aufgabe hat, unternehmensspezifische Standardaktivitäten und Standardschritte zu erstellen. Die Standardschritte enthalten Aktivitäten, die für diesen Arbeitsschritt immer enthalten sein müssen. Diese Standardisierung auf der Schrittebene muss auch auf der Prozess-Ebene möglich sein. Somit kann der Prozessadministrator gemeinsam mit dem verantwortlichen Sachbearbeiter einen Standardprozess definieren, der die nötigen Schritte und Aktivitäten enthält. Dieser Standardprozess dient als Basis für weitere zu modellierende Prozesse und enthält alle notwendigen Schritte und Aktivitäten eines fachbezogenen Standardprozesses. Besteht die Notwendigkeit, einen weiteren Prozess für diesen Fachbereich zu modellieren, so wird der Standardprozess herangezogen und für die weitere Bearbeitung eine Kopie in der Entwicklungsumgebung erstellt. Für eine detaillierte Beschreibung wird auf das Kapitel Prozessmodellierung verwiesen. Aus der oben beschriebenen Definition ergeben sich die folgenden Beziehungen zwischen den Bestandteilen eines Prozesses.



**Abbildung 1.9.** Beziehung zwischen Prozess und Schritt

## 1.9.2 Architektur

Die Beschreibung der Architektur ist an dieser Stelle sehr knapp gehalten, da in den folgenden Kapiteln noch ausreichend auf die Software-Architekturen eingegangen wird. Der Software-Architekt muss bei dem Entwurf der workflow-basierten Lösung darauf achten, dass bei der Anbindung von Applikationen möglichst standardisierte Technologien und unternehmenseinheitliche Anbindungen verwendet werden, um so die Wartbarkeit der Schnittstelle zu optimieren. Es ist darauf zu achten, dass eine komponentenbasierte Lösung angestrebt wird. Dies wird z.Zt. durch die gängigsten Technologien unterstützt und weist für die Realisierung

sierung einer prozessorientierten Lösung die größten Vorteile auf. Näheres wird in dem Kapitel Software-Architekturen beschrieben.

Nach Meinung des Autors ist die in Abbildung 1.6 dargestellte Anbindung zwischen Worklist und Applikationen nur in wenigen Produkten und Lösungen vorzufinden. Die Realisierung dieser Anbindung würde bedeuten, dass die Aktivitäten direkt aus der Worklist, also dem Workflow-Client, mit den Applikationen kommunizieren und somit die Workflow-Engine nur bedingt von dem aktuellen Status der Funktionsausführung etwas mitbekommt. Die Applikation würde lokal auf dem Client installiert sein und man hätte so aus der Sicht des Prozesses wieder einen „Fat Client“. Dies sollte in einer modernen Architektur nicht mehr vorzufinden sein und auch nur in Ausnahmefällen erlaubt werden. Die Nachteile überwiegen bei dieser Art der Anbindung von Applikationen. Die größten Nachteile sind die Wartbarkeit der Schnittstelle und der Applikation. Würde eine neue Version der Applikation auf den Clients installiert werden, so müsste zusätzlich auch eine neue Version der Schnittstelle installiert werden. Der daraus entstehende Aufwand ist erheblich. Es empfiehlt sich, diese Art der Anbindung nur bei sehr speziellen Clients zu verwenden, also nicht für die Standardbearbeitung von Prozessen, sondern z.B. für einen CAD-Client, der in einem Prozess mit eingebunden werden muss.

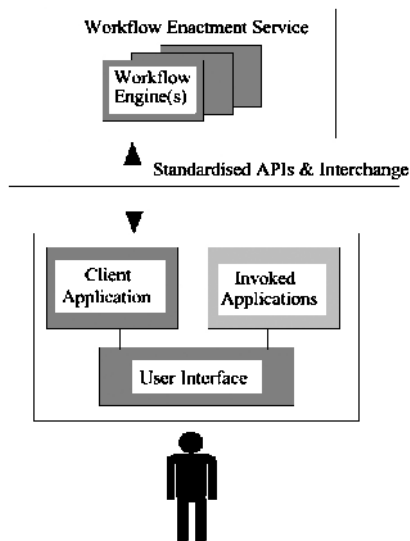
### 1.9.3 WfMC

Der Gedanke der WfMC ist prinzipiell zu befürworten. Nur ist aufgrund der hohen Anzahl von unterschiedlichen Interessen ein Standard, in dem z.B. die Komponenten und laufende Prozesse austauschbar sein sollen, illusorisch. Es ist nach ca. zehn Jahren nur für eine der fünf Schnittstellen eine sehr oberflächliche Definition verabschiedet worden, für eine weitere ist eine Verabschiedung geplant. Da die führenden Hersteller der Komponenten den definierten Standard auch nur sehr halbherzig verfolgen, ist in absehbarer Zeit keine Besserung zu erwarten. Die Definitionen sind demnach nur als Empfehlung zu sehen und können als Kriterien in Checklisten für die Evaluierung von WfMS verwendet werden. Auf der anderen Seite verwenden die Hersteller von WfMS und Modellierungstools sehr gerne den Begriff „WfMC-konform“. Kritisch betrachtet ist dies aufgrund der sehr abstrakten Definition relativ leicht zu erreichen. Genauer gesehen gelingt es nur mit erheblichem Aufwand, zwei Komponenten aus der WfMC-Architektur von unterschiedlichen Herstellern miteinander zu verknüpfen.

An dieser Stelle ist aber auch zu erwähnen, dass die Nennung „WfMC-konform“ letztendlich durch den Kunden erzwungen wurde, da dies mit als Auswahlkriterium für ein WfMS verwendet wird. Dadurch sehen sich die Hersteller in der Pflicht, solch eine Aussage zu treffen, obwohl der Standard bisher im Großen und Ganzen nur Theorie ist und im heutigen Zustand nur als Leitfaden gesehen werden kann.

Wie dem Leser wahrscheinlich aufgefallen ist, sind in Abbildung 1.5 und Abbildung 1.7 die Darstellungen der Schnittstellen etwas verwirrend. So wird in Abbil-

dung 1.5 zwischen der Workflow Client Application und der Invoked Application keine Schnittstelle dargestellt. In Abbildung 1.7 ist dagegen eine Verbindung zwischen dem Worklist Handler (ein Modul der Workflow Client Application) und weiteren Invoked Applications eingezeichnet. Eine Erklärung soll mit Hilfe der Abbildung 1.10 gegeben werden.



**Abbildung 1.10 . WfMC Workflow Client Application**

In dieser Abbildung wird dargestellt, dass die Applikationen (Invoked Applications) auf dem Client, laut der WfMC, in die Workflow Client Application mit einbezogen sind und der Datenaustausch direkt auf dem Client durchgeführt wird, ohne die Workflow-Engine mit einzubeziehen. Als Beispiel kann hier ein E-Mail-Client genannt werden, der auf Grund einer eingegangenen Mail ein Ereignis auslöst. Dieser Aufbau birgt nach Meinung des Autors eine große Gefahr, da generell die Daten ohne Einbeziehen der Workflow-Engine ausgetauscht werden können. Im Falle eines E-Mail-Clients kann das bedeuten, dass weder der Inhalt der Mails, noch das Eintreffen von E-Mails für die Workflow-Engine ersichtlich wird. In den aktuellen Software-Architekturen sollte darauf verzichtet werden, da hier zum einen die klaren Grenzen, die in Abbildung 1.5 dargestellt sind, zwischen dem WfMS und den angebundenen Applikationen aufgeweicht werden. Zum anderen findet die Kommunikation bzw. der Datenaustausch ohne die Workflow-Engine statt. Dieser Mangel sollte am Beispiel des E-Mail-Clients dadurch beseitigt werden, dass die Workflow-Engine direkt mit dem E-Mail-Server kommuniziert. Dadurch werden z.B. bei einer schlechten oder nicht permanenten Verbindung zwischen dem Workflow-Client und der Workflow-Engine die Aktionen korrekt mitprotokolliert und die eigentlichen Funktionalitäten in der Workflow-Engine auf dem Server realisiert. Ein solcher Aufbau entspricht dann eher einem Thin-Client, der in neuen Software-Architekturen auf jeden Fall angestrebt werden sollte.

## 2 Enterprise Application Integration (EAI)

Wie in der Einleitung beschrieben, beinhaltet das Thema *Workflow-based Integration* die beiden Themen Workflow und EAI. In diesem Kapitel wird ein Überblick über EAI gegeben, so dass es dem Leser leichter fällt, die Zusammenhänge der beiden Themen zu verstehen und die unterschiedlichen Varianten von EAI kennen zu lernen.

Es gibt mehrere Motivationen für die Notwendigkeit von Anwendungsintegration. Zum einen haben sich in der Vergangenheit unterschiedliche Applikationen und Datenbanken in den Unternehmen etabliert, die für die Abwicklung der Geschäftsprozesse benötigt werden. Im schlimmsten Fall werden diese Applikationen als Insellösungen betrieben und können keine Daten miteinander austauschen. Das hat nicht nur zur Folge, dass die Geschäftsprozesse träge bearbeitet werden, sondern dass auch eine hohe Fehlerrate bei der Bearbeitung auftritt. Der Ablauf eines Prozesses kann meistens nicht nachvollzogen werden. Eine zweite Motivation ist die Fusion von Unternehmen. Es müssen bei einer Fusion u.U. mehrere verschiedene IT-Umgebungen zusammengeführt werden, um eine einheitliche Datenbasis und durchgängig bearbeitbare Geschäftsprozesse zu erhalten. EAI ist zur Lösung des technischen Problems entstanden, dass unterschiedliche Applikationen in einem Unternehmen über verschiedene Protokolle oder technische Schnittstellen miteinander verknüpft werden müssen. Die Verknüpfung zwischen den Applikationen kann auf mehrere Arten realisiert werden, die sich im Aufwand und Komfort der Realisierung und der Pflege der Schnittstellen unterscheiden. Diese Varianten werden Im Folgenden beschrieben. Zuerst erfolgt aber eine Erklärung von EAI. Der Leser kann so beurteilen, welche Art von EAI er in seinem Unternehmen vorfindet oder einführen möchte. Weiterhin werden Hinweise für den korrekten Einsatz eines EAI-Produktes gegeben, welche immer in Verbindung mit den Kapiteln Software-Architekturen, Workflow und Workflow-based Integration (WfBI) stehen.

### 2.1 Begriffserklärung

Wie so oft in der IT gibt es zu EAI nicht nur eine Definition und zu den zugehörigen Begriffen nicht nur eine Erklärung. Für dieses Buch werden die folgenden Definitionen erläutert und verwendet.

Eine kurze Definition hat die *Gartner Group* veröffentlicht:

„Application integration means making independently designed systems work together.“

Nach einem Artikel von *David Linthicum* (Linthicum 1999) wird EAI wie folgt beschrieben:

“EAI is unrestricted data sharing among any connected applications or data sources in the enterprise. You need to share this data without making sweeping changes to the applications or data structures. “

Diese beiden Definitionen lassen sich wie folgt zusammenfassen:

EAI ist eine Software, durch die das Unternehmen die Möglichkeit erhält, bestehende und neue Applikationen miteinander zu verknüpfen, damit auf der Basis gemeinsamer Daten und Funktionen die Geschäftsprozesse abgewickelt werden können. Durch die Verwendung eines EAI-Produktes ist es möglich, die in einem Geschäftsprozess eingebundenen Applikationen in einer heterogenen IT-Umgebung zu verwenden und somit einen Datenaustausch zwischen den Applikationen ohne einen Medienbruch zu realisieren.

Um EAI besser verstehen und die EAI-Produkte besser einordnen zu können, werden hier die unterschiedlichen Formen von EAI dargestellt.

### **2.1.1 Formen von EAI**

In Anlehnung an die *Gartner Group* werden folgende Ausprägungen von EAI betrachtet:

- Application-to-Application (A2A)
- Business-to-Business (B2B)
- Business-to-Consumer (B2C)

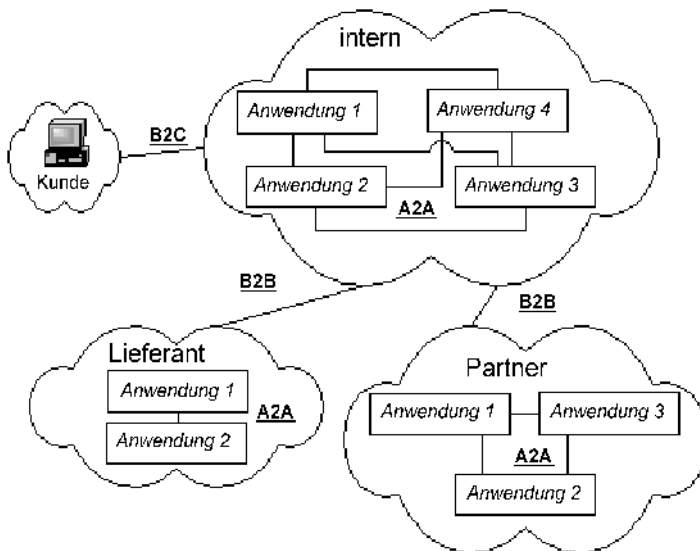
Eine Unterscheidung der drei Formen betrifft hauptsächlich die Tragweite der realisierten Geschäftsprozesse und die eingebundenen Applikationen. Werden Applikationen integriert, welche nur innerhalb eines Unternehmens verwendet werden, so spricht man von einer *A2A-Integration*. Bei dieser Form der Integration liegt der Schwerpunkt darin, zuerst die Applikationen für die strategischen Geschäftsprozesse miteinander zu verbinden, um so einen effizienten Ablauf der Prozesse zu erreichen. Durch eine entsprechende Architektur bzw. ein EAI-Produkt wird dies innerhalb der Unternehmensgrenzen realisiert.

Wird aufgrund von Geschäftsbeziehungen über die Unternehmensgrenzen hinweg z.B. das Internet als weitere Geschäftsplattform verwendet, spricht man von einer *B2B-Integration*. Eine solche Form der Integration setzt eine sichere Architektur

bezüglich Datenzugriff auf der einen und Investitionssicherheit auf der anderen Seite voraus. Das bedeutet, dass hier ein besonders sicherer und stabiler Standard für die Kommunikation zwischen den Unternehmen eingesetzt werden muss: nicht nur die eigenen Applikationen sind betroffen, sondern auch die der Geschäftspartner. Diese Form der Integration ist die aufwändigste, da häufig sowohl das eigene Unternehmen als auch die Geschäftspartner eine hohe Anzahl von Applikationen verwenden, die es zu integrieren gilt.

Die letzte Form der Integration ist die *B2C-Integration*. Dabei wird der Endkunde in den Geschäftsprozess integriert. Dies kann relativ einfach durch das Bereitstellen eines browserbasierten Web-Clients realisiert werden, da der Endkunde in der Regel keine eigenen Applikationen verwendet, die den Gesamtprozess unterstützen. Der Endkunde wird so mit geringem Aufwand als Anwender in die Prozesse integriert.

Schaut man sich die angebotenen EAI-Produkte an, so kann man erkennen, dass diese primär für B2B und A2A verwendet werden können. Die B2C-Integration kann im Allgemeinen durch einen weiteren Zugang (Channel) mit einfachen Mitteln erreicht werden.



**Abbildung 2.1.** EAI-Formen

Die oben beschriebenen Formen können anhand der Integrationsart der Realisierung weiter untergliedert werden. Die Integrationsarten unterscheiden sich darin, wie eng bzw. umfangreich die Integration realisiert wurde bzw. realisiert werden soll. Alle drei Integrationsformen (A2A, B2B und B2C) können unter Verwendung der folgenden Integrationsarten umgesetzt werden.



### 2.1.2 Integrationsarten

Diese Untergliederung beginnt mit der einfachsten und somit auch der kostengünstigsten Integrationsart.

#### Datenintegration

Mit der Datenintegration wird erreicht, dass alle in den Geschäftsprozessen verwendeten Applikationen mit demselben Datenbestand arbeiten und die benötigten Daten über ein festgelegtes Format ausgetauscht werden. Dies kann beispielsweise durch eine Dateischnittstelle realisiert werden. Jede Applikation erzeugt eine Exportdatei, die von anderen Applikationen importiert werden kann. Mögliche Datenformate für die Dateien können das CSV-Format oder XML-Format sein. Es ist aber auch denkbar, dass die Applikation A, welche die Daten benötigt, direkt auf die Datenquelle B (Datenbank) der Applikation B zugreift. Dies wird durch das Erteilen einer Leseberechtigung für erforderliche Datenansichten (z.B. Views) realisiert. Das Schreiben der Anwendung A in die Datenquelle B ist technisch möglich, kann aber zu inkonsistenten Datenbeständen führen, wenn nicht die Schreibfunktionen der Anwendung B verwendet werden. Eine weitere Variante ist, dass eine Anwendung die Daten aus mehreren Datenquellen (z.B. Datenbanken) bezieht und dies durch eine Zugriffsschicht realisiert wird. Die Zugriffsschicht vereinfacht den Zugriff auf mehrere Datenbanken, da es für die Anwendung so aussieht, als ob nur eine Datenbank vorhanden ist. In Abbildung 2.2 und Abbildung 2.3 sind die beiden zuletzt genannten Möglichkeiten der Datenintegration dargestellt.

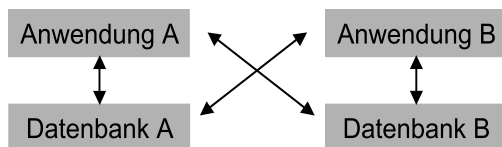


Abbildung 2.2. Datenintegration mit zwei Anwendungen

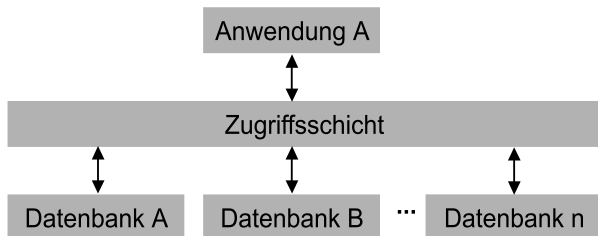


Abbildung 2.3. Datenintegration mit einer Anwendung

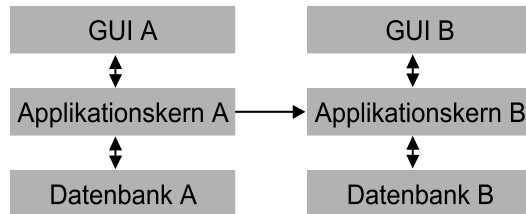
#### Funktionsintegration

Eine weitere Integrationsart ist die Funktionsintegration. Diese Art ist in der Realisierung aufwändiger als die Datenintegration und setzt voraus, dass die Applikationen ein *Application Programming Interface* (API) zur Verfügung stellen. Diese

Interfaces ermöglichen es, dass die Funktion aus Anwendung A die Funktion der Anwendung B ausführt. So kann erreicht werden, dass für den Prozess erforderliche Funktionalitäten nicht doppelt realisiert werden müssen. Weiterhin kann ein Funktionsaufruf einer Applikation entsprechende Funktionsaufrufe in weiteren Applikationen verursachen. Dies können folgende Applikationen sein:

- Standardanwendungen
- Datenbanken
- Legacy-Anwendungen
- Systeme von Geschäftspartnern

Betrachtet man den Begriff der „Funktion“ näher, so ist es erforderlich, dass man von der herkömmlichen Gliederung in Anwendungen Abstand nimmt. Dies bedeutet, dass man eher in fachlichen Funktionspaketen denken muss und daher der Begriff „Komponente“ treffender ist. Wird angestrebt, mehrere große Anwendungen miteinander zu verbinden, so ist beim Design der Lösung auch darauf zu achten, wie die einzelnen Komponenten verteilt und so z.B. auf unterschiedlichen Servern betrieben werden können. Man spricht dann von verteilten Funktionalitäten in unterschiedlichen Adressräumen. Dies ist nicht durch eine im herkömmlichen Sinne genannte API möglich, sondern muss durch die Verwendung von CORBA, COM/DCOM (vgl. Kapitel Software-Architekturen) oder anderen komponentenbasierten Technologien realisiert werden. In Abbildung 2.4 wird eine Integration über Funktionsaufrufe dargestellt.



**Abbildung 2.4.** Funktionsintegration

### Komponentenintegration

Die Komponentenintegration kann auf Basis von COM/DCOM, CORBA oder EJB-Containern (J2EE) realisiert werden. Unter einer Komponente soll in diesem Buch ein Paket an Funktionen verstanden werden, die aufgrund von fachlichen Anforderungen zusammengefasst und durch einen in sich abgeschlossenen Programmcode realisiert wurden. Komponenten können in mindestens drei Kategorien unterteilt werden: Plug-in, Glueware und Container. Bei einem Plug-in handelt es sich um eine abgeschlossene Funktionalität, welche in eine Anwendung eingebettet wird. Als Beispiele können ein PDF-Reader oder ein MPEG-Player genannt werden, die in einen Browser integriert sind. Für ein Plug-in muss die Schnittstelle sehr genau definiert sein, damit alle Funktionen und Nachrichten (Messages) richtig interpretiert werden können.

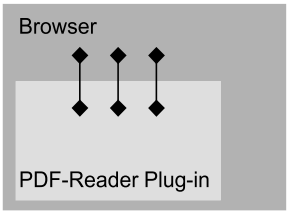


Abbildung 2.5. Komponentenintegration – Plug-in

Gegenüber einem Plug-in handelt sich bei Glueware um die Integration von Komponenten, die z.B. durch einen EAI-Integrationsserver verbunden werden. Der Integrationsserver verwaltet alle Daten und Funktionsaufrufe, die durch die Anwendung oder eine eigene Benutzeroberfläche angefordert werden. Als Verbinder (Glue = Klebstoff) wird hier meistens eine Skriptsprache verwendet.

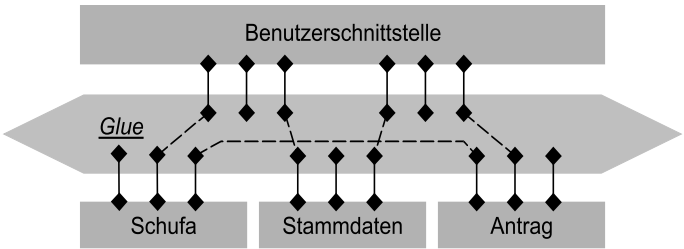


Abbildung 2.6. Komponentenintegration – Glue

Die dritte Kategorie ist der Container. Ein Container kann verschiedene Komponenten für die Darstellung (z.B. JSP) und für die fachlichen Funktionen (z.B. EJB) beinhalten. In Abbildung 2.7 sind die EJBs zu erkennen, die für die Stammdatenbearbeitung, „Schufa-Abfrage“ und die Bearbeitung der Antragsdaten in einem

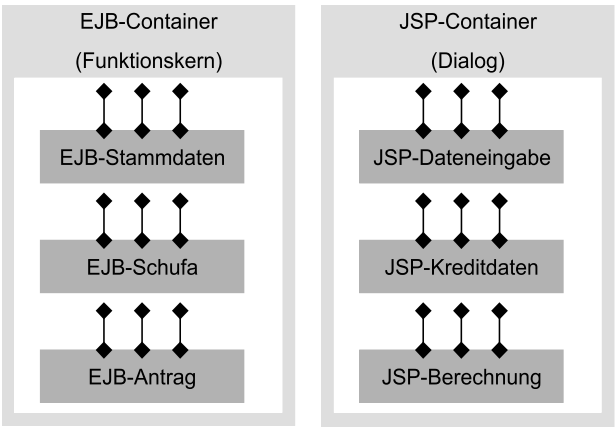


Abbildung 2.7. Komponentenintegration – Container

EJB-Container zur Verfügung gestellt werden. Um die Daten eingeben zu können, werden in dem JSP-Container die notwendigen Oberflächen bereitgestellt, die die Methoden in den EJBs verwenden können. Die gebräuchlichsten Container findet man z.B. als zentrales Konzept in der J2EE-Architektur; diese wird im Kapitel Software-Architekturen beschrieben.

### Prozessintegration

Die aufwändigste Integrationsart ist die Prozessintegration. Bei dieser Art der Integration durchläuft der Prozess die benötigten Applikationen und die Prozesssteuerung (WfMS) ruft die Funktionen in den Applikationen auf. Dies kann durch zwei Varianten des Benutzerdialoges realisiert werden. In der einen Variante wird das WfMS für den Anwender nicht sichtbar, d.h. es steuert die Anwendungen durch den definierten Prozess und ruft die Dialoge aus den Anwendungen auf. Die andere Variante ist, dass das WfMS die Dialoge über eine entsprechend integrierte Oberfläche (Tätigkeitenliste, Aktivitätenliste) zur Verfügung stellt und die Funktionen in den Anwendungen mit den erforderlichen Parametern aufruft. Dies hat den Vorteil, dass das look-and-feel der Benutzerschnittstelle einheitlich ist. Die Realisierung bzw. Steuerung von Teilprozessen in den Applikationen, welche sowohl die Daten als auch den Status des übergeordneten Prozesses an die nachfolgenden Applikationen weitergeben, kann eine weitere Variante sein. Diese soll aber aufgrund der Komplexität hier nicht weiter verfolgt werden.

Eine effektive Prozessintegration kann nicht ohne eine geeignete Software-Plattform realisiert werden, da die Steuerung der Prozesse und die Datenübergabe an die am Prozess beteiligten Applikationen durch eine einheitliche Schnittstelle zu realisieren ist. Eine solche Schnittstelle sollte zumindest im jeweiligen Unternehmen standardisiert sein.

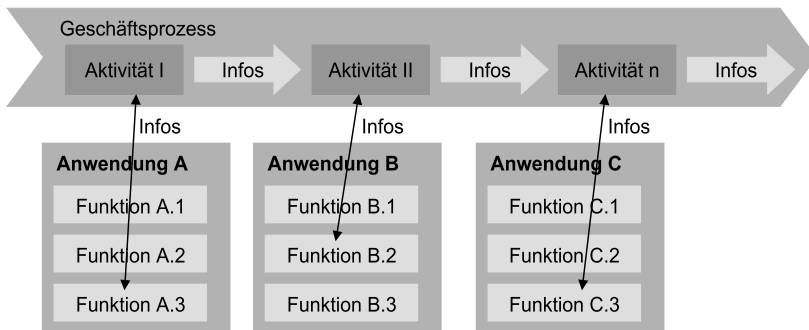
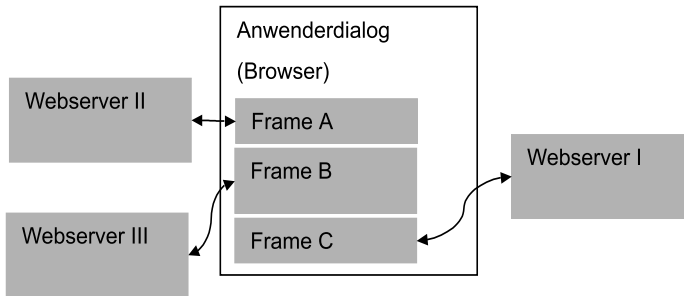


Abbildung 2.8. Prozessintegration

### Dialogintegration und Benutzerschnittstellenintegration

Durch die Verbreitung der Web-Technologien hat die Dialogintegration oder auch Benutzerschnittstellenintegration wieder eine größere Bedeutung erlangt. Durch sie werden Benutzereingaben funktional getrennt und in separaten Bereichen (Frames) zusammengefasst. Diese funktional getrennten Frames werden mit anderen

Frames in einem Dialog dargestellt. Der große Nachteil dieser Integrationsart ist die u.U. sehr aufwändige Synchronisierung der Daten zwischen den Frames.



**Abbildung 2.9.** Dialogintegration über Frames

## 2.2 Entwicklung von EAI

Enterprise Application Integration entwickelte sich mit den Anforderungen, die eine sich ändernde IT-Infrastruktur vorgab. Die Anfänge liegen in den groß- und mittelständischen Unternehmen, als zum zentralen Großrechner weitere DV-Anlagen dazukamen.

### 2.2.1 Innerhalb der Unternehmen

Mittelständische und große Unternehmen haben seit den Anfängen der EDV mit einer zentralen IT-Plattform z.B. in Form eines Zentralrechners (Host, Mainframe) gearbeitet und dies bis zum Ende der 80er Jahre erfolgreich weitergeführt. In einem solchen homogenen IT-Umfeld war die Integration von weiteren Anwendungen relativ einfach zu realisieren, da auf demselben Rechner, mit demselben Betriebssystem und unter Umständen auch mit denselben Compilern für die Programmiersprache Cobol gearbeitet wurde. Ende der 80er begann der Einzug der PCs in die Geschäftswelt. Dadurch kamen neue Betriebssysteme und Netzwerktopologien zu der bestehenden Infrastruktur hinzu. Die Folge war ein IT-Umfeld, das zunehmend inhomogen und dezentralisiert wurde. Es begannen sich die Nachteile eines solchen IT-Umfeldes herauszukristallisieren. Neben den Vorteilen wie z.B. schnelle und flexible Umsetzung von Fachbereichsanforderungen und einer höheren Ausfallsicherheit bezüglich der Prozessbearbeitung im Unternehmen (die Prozesse konnten teilweise auch ohne den Host bearbeitet werden) ergaben sich auch Nachteile wie redundante Datenhaltung, Probleme der Datensynchronisierung und Prozesse, die sich über mehrere Systeme erstreckten. Die Anbindungen zwischen den Systemen waren aber nicht-technischer Art, die Prozesse durchliefen mehrere Medienbrüche. Ende der 80er und Anfang der 90er Jahre kamen Technologien auf den Markt, die es ermöglichten, zwischen den Systemen eine technische Schnittstelle zu realisieren. Mit Sockets, Remote Procedure/Function Calls (RPC/RFC) und CORBA standen Technologien zur Verfügung, die oben genann-

ten Probleme zu minimieren. Mit diesen Technologien und einer immer schneller werdenden Hardware begann eine neue Art der Anwendungsentwicklung. Durch die komponentenbasierte und objektorientierte Software-Entwicklung wurde es möglich, Applikationen schneller zu entwickeln und in der heterogenen IT-Umgebung zu verteilen. Aufgrund der Vielzahl von Applikationen und Systemen in den Unternehmen entstand so eine große Anzahl von Schnittstellen, die neben dem hohen Realisierungsaufwand auch einen erheblichen Wartungsaufwand zur Folge hatten. In großen Unternehmen ist man heute nicht mehr in der Lage, alle Verbindungen (Schnittstellen) zu den Applikationen, die in der Regel eine Punkt-zu-Punkt-Verbindung sind, zu pflegen. Man behandelt daher lieber ganze Systeme und auch Applikationen als „Blackbox“: Es werden die internen Funktionen so gelassen, wie sie sind, und man versucht, Erweiterungen um diese Boxen zu realisieren. Die Folge einer solch aufwändigen „gewachsenen“ Architektur wird als Schnittstellenspagetti (Gartner Group) bezeichnet und sollte durch die Einführung einer Message Oriented Middleware (MOM) behoben werden. Mit der MOM werden die Daten in Pakete verpackt und in Form von Nachrichten (Messages) an eine Middleware übergeben. Für den Transport an den oder die richtigen Empfänger der Nachrichten ist die Middleware verantwortlich. Somit ist der Aufwand für eine Integration einer neuen Applikation sehr gering, da der Programmierer sich nur um das Senden und das Empfangen der Daten kümmern muss. Ein gravierender Nachteil besteht allerdings darin, dass in bestehende Applikationen eingegriffen werden muss, die MOM nicht unterstützen. Solche Eingriffe wurden durch die Entwicklung des Message Brokers minimiert. Die Hersteller der Message Broker erlauben durch die Realisierung von Adaptern (Connectoren) die Anbindung und

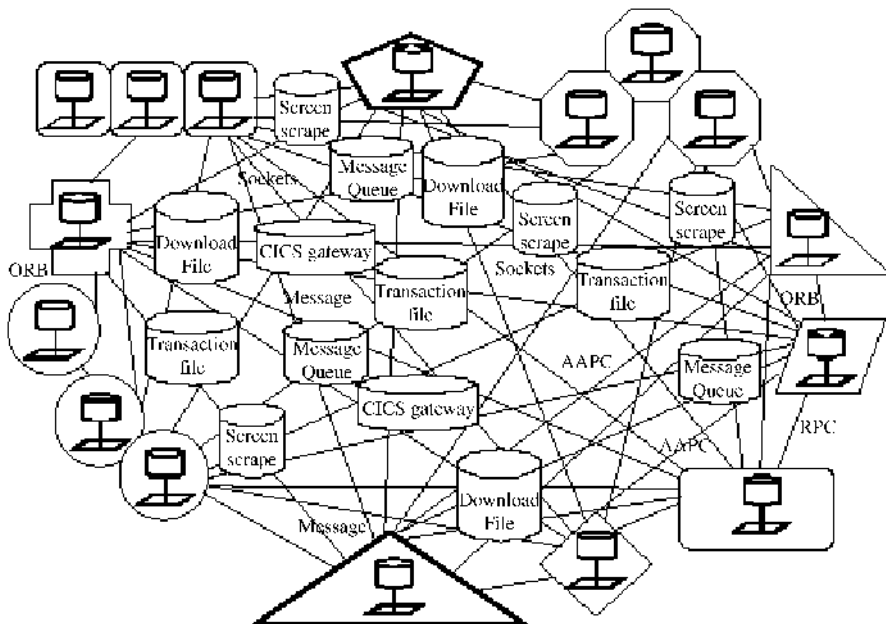


Abbildung 2.10. Schnittstellenspagetti nach Gartner

somit den Datenaustausch an einen so genannten Integrationshub. Diese Adapter können durch die Applikationshersteller oder durch Eigenentwicklungen realisiert werden. Die Voraussetzung für eine Eigenentwicklung ist ein vorhandenes applikationsabhängiges *Software Developer Kit* (SDK), um Adapter selber realisieren oder bestehende Adapter erweitern zu können. Der Message Broker wird so zur zentralen Stelle, vom dem alle Daten (Nachrichten) empfangen und verwaltet werden.

### 2.2.2 Zu anderen Unternehmen

Zusätzlich zu dem Austausch der Daten innerhalb eines Unternehmens wurde es immer notwendiger, Daten zwischen Unternehmen auszutauschen. Ein gutes Beispiel sind hier die Banken, die ihre Transaktionen austauschen müssen. Aus dieser Notwendigkeit schlossen sich Unternehmen zusammen und definierten das Electronic Data Interface (EDI). Auf Basis des Value Added Network (VAN) musste jedes Unternehmen ein Gateway implementieren, das der Spezifikation des VANS entsprach. In der Folge ergaben sich weitere Standards, die sich branchen- und länderspezifisch entwickelten. Der Bekannteste ist der allgemeine UN-Standard EDIFACT. Mit der Verbreitung des Internets und den damit verbundenen neuen Technologien war es auch solchen Unternehmen möglich, sich an dem Datenaustausch zu beteiligen, die sich bisher aus Kostengründen nicht an dem VAN beteiligten. Ende der 90er waren die Technologien vorhanden, die einen sicheren Datenaustausch zwischen den Unternehmen über das Internet gewährleisten. Die daraus entstandenen Produkte bzw. Technologieplattformen basieren heute überwiegend auf den Technologien J2EE und DCOM (heute .NET) und werden in den folgenden Kapiteln beschrieben.

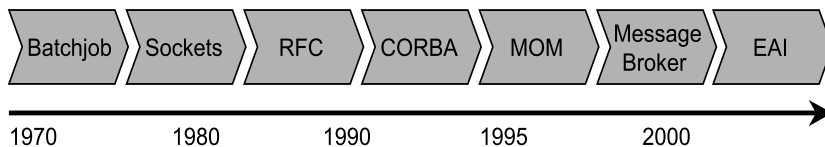


Abbildung 2.11. Historie von EAI

## 2.3 Eigenschaften eines Integrationsservers

Unter einem Integrationsserver versteht man das Software-Paket (EAI-Produkt) eines Herstellers, welches als Plattform für die Integration der Applikationen in einem Unternehmen und zwischen Unternehmen dient. Integrationsserver unterscheiden sich nicht nur durch verschiedene Architekturen, sondern auch in ihren Eigenschaften. Der Leser sollte bei der Auswahl eines Integrationsservers auf diese im Folgenden aufgeführten Eigenschaften achten, da sie die Architektur des eigenen IT-Systems beeinflussen und auch auf die bestehenden Applikationen Auswirkungen haben.

### Kommunikationsarten

Die Kommunikationsarten unterscheiden sich grundlegend in synchrone und asynchrone Kommunikation.

Bei der synchronen Kommunikation wartet der Sender so lange, bis der Empfänger ihm den Empfang der Nachricht bestätigt oder eine Antwort gesendet hat. Die synchrone Kommunikation kann in verschiedenen Kommunikationsmustern wie Request/Reply, One-Way oder Polling realisiert werden.

Bei dem Request/Reply-Muster verschickt ein Sender eine Nachricht in Form einer Anfrage an einen Empfänger und wartet so lange mit der Fortführung der Kommunikation, bis er eine Antwort vom Empfänger bekommt. Das One-Way-Muster basiert auf dem Reply/Request-Muster, hier erhält der Sender sofort nach Eingang der Anfrage eine leere Nachricht als Antwort, dass die Anfrage angekommen ist. Das Polling unterscheidet sich von dem Reply/Request-Muster dadurch, dass der Prozess sobald der Sender die Anfrage verschickt hat, weiterverarbeitet und in zuvor bestimmten Intervallen kontrolliert wird, ob eine Antwort angekommen ist.

Bei Request/Reply besteht die Gefahr, dass sich das System komplett selber blockiert, falls eine der beteiligten Applikationen aus irgendeinem Grund nicht antworten kann. Dieser gravierende Nachteil entfällt bei One-Way und Polling, da der Sender nicht auf eine Antwort des Empfängers wartet.

Bei der asynchronen Kommunikation schickt der Sender eine Nachricht und arbeitet seine weiteren Aufgaben oder Nachrichten ab. Der Empfänger sendet zu einem nicht vorgegebenen Zeitpunkt eine Antwort oder Bestätigung. Die asynchrone Kommunikation unterscheidet sich in die Kommunikationsmuster Message Passing, Publish/Subscribe und Broadcast.

Beim Message Passing dienen Warteschlangen als Puffer zwischen Sender und Empfänger. Da der Absender nicht auf eine Antwort wartet, kann er mit seinem Prozess fortfahren. Mit dem Publish/Subscribe-Muster werden Nachrichten nur an solche Empfänger versendet, die sich zuvor über ein Abonnement an den Absender angemeldet haben. Beim Broadcasting werden Nachrichten an alle Empfänger gesendet, die sich im Netzwerk befinden und dann selbst entscheiden, ob diese Nachricht für sie relevant ist oder nicht. Der Unterschied zum Publish/Subscribe liegt darin, dass die Empfänger sich bei dem Absender anmelden müssen, um die Nachricht zu erhalten.

Anhand der Kommunikationsart kann noch keine Aussage über die Qualität der Nachrichtenübermittlung getroffen werden. Um sicherzustellen, dass die Nachrichten übermittelt werden, kann eine Kombination aus einem Muster der garantierten Auslieferung und der asynchronen Kommunikation gewählt werden. In diesem Fall werden alle Nachrichten protokolliert und dauerhaft vorgehalten. Dies wird unabhängig von dem Übermittlungsdienst durchgeführt, um zu vermeiden, dass die Nachrichten noch zur Verfügung stehen, falls der Übermittlungsdienst



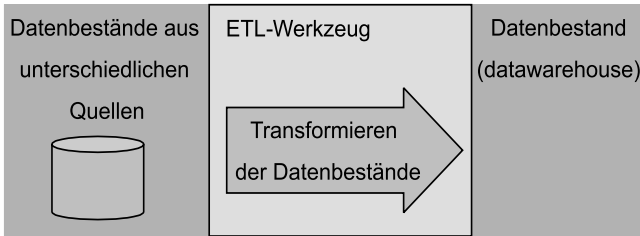
ausgefallen ist. Anhand einer vordefinierten Zeit (Timeout) wird kontrolliert, ob der Empfänger sich bereits gemeldet hat. Ist dies nicht der Fall, so wird die Nachricht nochmals gesendet. Dies wiederholt der Übermittlungsdienst so lange, bis er eine positive Antwort durch den Empfänger erhalten hat. Diese Verfahren können in unterschiedlichen Qualitätsstufen konfiguriert werden. Mögliche Parameter sind Länge des Timeouts, Anzahl der Wiederholungsversuche pro Zeiteinheit, Priorisierung der Nachrichten und Detaillierungsgrad der Protokollierung.

### **Adapter**

Als Adapter bezeichnet man einzelne Module, die entweder fertig durch den Produkthersteller der Fachanwendung beziehungsweise den Hersteller des Integrationservers zur Verfügung gestellt oder durch eine Eigenentwicklung mit einer Programmierungsumgebung realisiert werden. Es ist wichtig, dass der Integrationsserver alle Adapter vorweisen kann oder die Möglichkeit bietet, diese für die Anbindung bereits bestehender Applikationen programmiertechnisch zu realisieren. Die Integrationsserver unterscheiden sich nicht nur in der Anzahl und den Typen von Adaptern, sondern auch in der Höhe des Aufwandes, mit dem weitere Adapter realisiert werden können. Hierfür bieten die Hersteller unterschiedliche Programmiersprachen und Frameworks an, die die Realisierung von kundenspezifischen Adaptern vereinfachen.

### **Nachrichtenformate**

Um die Kommunikation zwischen Client und Server sowie Server und Server (wie oben beschrieben) sicherzustellen, muss das Format der Nachrichten für den Informationsaustausch festgelegt sein. Dies wird in den meisten Fällen durch die verwendete Technologie und somit durch den Hersteller des EAI-Servers bestimmt. Es gibt Basisformate, welche in den RPC-basierten Servern wie DCS und CORBA realisiert wurden. Diese wurden aber nur auf einer sehr abstrakten Ebene spezifiziert. Das hat zur Folge, dass in einem Unternehmen das Format im Detail definiert werden muss. Falls es zu einer Kommunikation mit einem Altsystem oder einem Unternehmen mit einem anderen Datenformat kommt, können Werkzeuge zur Datentransformation verwendet werden, um so die Kommunikation in dem erforderlichen Format zu ermöglichen. Das beste Beispiel für ein Nachrichtenformat ist das sich immer mehr durchsetzende XML-Format. Es ist zu berücksichtigen, dass XML nur das Format definiert und die Nachrichten nicht normiert. Deshalb müssen zentral im Unternehmen das Vokabular und die Dokumentenformate festgelegt werden (vgl. Glossar). Kommt es zu einer Kommunikation zwischen weiteren Unternehmen, ist es von Vorteil, wenn es eine zentrale Stelle gibt, in der diese Nachrichten gemeinsam normiert werden. Da dies bisher noch nicht in ausreichendem Maße stattgefunden hat, kann man sich auch hier der entsprechenden ETL-Werkzeuge (Extraktions-, Transformations- und Ladewerkzeug) bedienen, die eine Datentransformation durchführen. Solche ETL-Werkzeuge sind schon seit längerem auf dem Markt und können die Nachrichten in das gewünschte Zielformat konvertieren, das für die Kommunikation zwischen den EAI-Servern und den Altsystemen benötigt wird.



**Abbildung 2.12.** ETL-Werkzeug

### Sicherheit

Mit dem Begriff Sicherheit ist hier der Zugriff auf Daten und die Übertragung von Nachrichten gemeint und nicht die Ausfallsicherheit oder Verfügbarkeit. Da ein EAI-Server mit vielen Applikationen des Unternehmens in Verbindung steht, ist das Thema Sicherheit von großer Bedeutung. Allerdings liegt kein standardisiertes Sicherheitskonzept für einen EAI-Server vor, daher müssen die Architekten und Administratoren ein eigenes Sicherheitskonzept erarbeiten. Im Folgenden werden die Möglichkeiten aufgezählt, die ein EAI-Server zur Verfügung stellen muss.

Grundsätzlich hat sich jeder Anwender und jedes fremde System gegenüber dem EAI-Server zu authentifizieren, um dessen Berechtigung zum Datenzugriff zu verifizieren. Die Daten für die Authentifizierung können in einem LDAP-Directory (Lightweight Directory Access Protocol) zur Verfügung gestellt werden.

Neben der oben genannten Möglichkeit findet man in der Praxis Sicherheitskonzepte, die als Ergänzung zu den in den Produkten realisierten Varianten zu sehen sind. Diese Konzepte kann man wie folgt unterteilen.

**Sicherheitskonzepte der eingebundenen Technologien:**

Basiert der Integrationsserver z.B. auf CORBA, so wird der Sicherheitsdienst von CORBA verwendet.

**Sicherheitskonzepte der EAI-Produkte:**

Zusätzlich zu dem Sicherheitskonzept der Basistechnologie wird das Sicherheitskonzept der Lösung durch das Sicherheitskonzept des EAI-Produktherstellers erweitert.

**Authentifizierung und verschlüsselte Kommunikation:**

Die Kommunikation über Adapter wird verschlüsselt, um so eine sichere Kommunikation zwischen zwei Systemen zu realisieren. Zusätzlich besteht die Möglichkeit, dass sich ein Adapter bei einem Fremdsystem authentifizieren kann.

Adapter an andere Sicherheitssysteme:

Der EAI-Server beinhaltet Funktionalitäten, um sich gegenüber anderen Sicherheitssystemen zu authentifizieren. Dies wird durch Standards wie RACF (Resource Access Control Facility) oder Zertifikate realisiert.

### **Routing**

Die Nachrichten enthalten immer die Adressaten. Würde man für den Adressaten immer einen physischen Namen wie z.B. eine IP-Adresse und den Port verwenden, wäre der Nachteil, dass neben einer aufwändigen Wartbarkeit des Systems auch die Lesbarkeit der Nachrichten für den Entwickler schlechter wäre. Deshalb verwendet man in den Integrationsservern einen Namensdienst (s.u.), welcher auf der grundlegenden Technologie (z.B. CORBA) basiert oder selbst durch den Hersteller realisiert wurde.

### **Repository**

Da die Basis eines Integrationsservers die Nachrichtentypen und deren Formate sind, besteht die Notwendigkeit, ein Repository (= Nachschlagewerk, Definitionsverzeichnis) für die Nachrichten und Formate in einem Integrationsserver zu implementieren. Mit diesem Repository werden die Nachrichten und Formate in den verschiedenen Versionen verwaltet und ausgewertet.

### **Namensdienst (Name service)**

Ein Namensdienst konvertiert technische Bezeichnungen für Nachrichtenziele und Objekte. Der Anwender, der Architekt oder der Administrator kann Namen für Systembestandteile (Client, Server, Gateway, ...) vergeben. Der Namensdienst wandelt den Namen in eine physische Adresse um und umgekehrt. Die Lesbarkeit/Nachvollziehbarkeit für den Menschen wird so erhöht. Aus den Beispielen eines Servernamens und einer E-Mail-Adresse ist dies ersichtlich. Der Servername, also der Name, den man verwenden kann, um den Server über das Netzwerk anzusprechen, kann *BRUECKE* lauten. Es ist aber auch möglich den Server über die IP-Adresse (= physischer Name) anzusprechen. Genauso bei der E-Mail-Adresse. Die E-Mail-Adresse des Empfängers, die der Anwender des Mailprogramms eingibt, hat immer das folgende Grundformat: *Empfaengername@domaenennamenname.suffix*. Diese Adresse wird von einem Namensdienst in eine physische Adresse übersetzt und ist u.U. ein Blatt in einem Verzeichnisbaum. Diese Strukturen werden z.B. in einem LDAP-Verzeichnisdienst oder in einem X.500-Verzeichnis abgelegt. CORBA, welches im Kapitel Verteilte Anwendungen mit CORBA noch näher beschrieben wird, enthält einen Namensdienst. Bei der Evaluierung bzw. beim Verknüpfen von Integrationsservern ist darauf zu achten, dass dieselbe Art von Namensdienst verwendet wird, da kein Standard in den Integrationsservern realisiert ist.

### **Load Balancing**

In diesem Abschnitt wird nicht nur auf das Load Balancing eingegangen, sondern auch auf Fail-over, Skalierbarkeit und Performance. Diese Eigenschaften liegen bei der Entscheidung für eine Systemarchitektur bzw. ihrem Design sehr dicht zu-

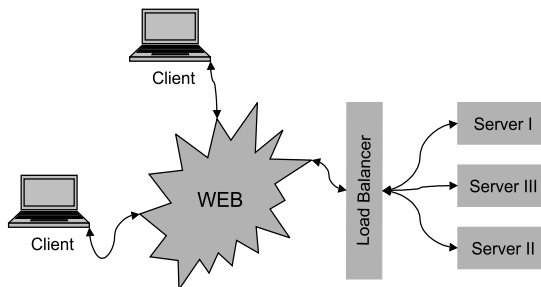
sammen bzw. sind auch voneinander abhängig. Bearbeitet das Unternehmen kritische Geschäftsprozesse, welche eine hohe Verfügbarkeit des Systems erfordern und einen großen Durchsatz an Daten aufweisen, ist es wichtig, ein leistungsfähiges und ausfallsicheres System zu realisieren. Dies kann aufgrund der konkreten Anforderungen sehr unterschiedlich in der Architektur und damit dem finanziellen Aufwand ausfallen. Es werden an dieser Stelle zwei verschiedene Verfahren für eine Fail-over-Strategie aufgeführt, um den unterschiedlichen Aufwand darzustellen und aufzuzeigen, welche Unterschiede dabei in den Bereichen Performance und Datensicherheit entstehen können.

### Standby

Die einfachste Variante ist, ein redundantes System mit derselben IP-Adresse neben dem primären System aufzustellen. Im Falle eines Server-Ausfalles wird das sekundäre System aktiviert. Ein entscheidender Nachteil für eine solche Lösung besteht darin, dass getätigte Transaktionen, welche auf dem primären System gestartet wurden, verloren gehen und eine Ausfallzeit in Kauf genommen werden muss. Außerdem sind die Kosten für Hardware und Wartung relativ hoch.

### Hot Standby

Das System wird ähnlich aufgebaut wie beim Standby-Verfahren. Es wird nur zusätzlich noch ein Load Balancer („Lastverteiler“) zwischen den Client und die Server geschaltet. In Abbildung 2.13 ist ein Beispiel mit drei Servern zu sehen, um so bei hohem Durchsatz die Anfragen zu bearbeiten und ein hochverfügbares und leistungsfähiges System zu erhalten. In dieser Konfiguration sind alle drei Server in Betrieb und werden durch den Load Balancer mit Anfragen versorgt. Fällt nun ein System aus, können die beiden anderen Systeme die Anfragen übernehmen und so eine hohe Ausfallsicherheit gewährleisten. Je nach Anforderungen der Lösung können folgende Technologien genutzt werden: Network Address Translation, DNS Based Load Balancing und Multi Node Load Balancing.



**Abbildung 2.13.** Hot-Standby

Da sich diese Lösungen auch sehr stark in den finanziellen Realisierungs- und Wartungsaufwänden unterscheiden, ist sehr genau zu prüfen, welche Anforderungen an das System gestellt werden. Das Ergebnis kann auch ein Kompromiss aus verschiedenen Verfahren und die bereits in den Integrationsserver integrierte Technologie sein.

**Monitoring**

Unter dem Begriff Monitoring versteht man in diesem Zusammenhang das Beobachten des Systems, insbesondere der Schnittstellen. Es werden Integrationsserver angeboten, die diesen Mechanismus bereits implementiert haben. So muss man nicht die u.U. bereits für das heterogene Netzwerk installierten Überwachungswerkzeuge wie OpenView oder Tivoli verwenden, sondern kann primär auf die mitgelieferten Monitoring-Werkzeuge des Herstellers zurückgreifen. Es ist ein großer Vorteil, wenn die Werkzeuge auch eine aktive Komponente besitzen: sie melden sich selber durch einen Alarm bei dem Administrator, falls die Schnittstelle nicht mehr einwandfrei funktioniert.

**Recovery**

Der Begriff Recovery (Wiederherstellung), der aus dem Bereich der Datenbanken kommt, beschreibt, wie eine „abgestürzte“ Datenbank wieder in einen konsistenten Zustand gebracht werden kann. Neben einem fehlerfreien Neustart der Datenbank muss auch sichergestellt werden, dass die Transaktionen zwischen Datenbank und Anwendung entweder komplett abgeschlossen oder gar nicht ausgeführt werden. Dieses Verfahren kann auch auf einen Integrationsserver angewendet werden. Das korrekte Verhalten der Transaktionen wird aber nicht alleine durch den Server sichergestellt. Auch der Entwickler hat durch das richtige Setzen des Anfangs und des Endes einer Transaktion sicherzustellen, dass die bearbeiteten oder eingegebenen Daten im Fehlerfall konsistent verfallen oder gespeichert werden. Neben dem korrekten Abschluss von Transaktionen ist auch darauf zu achten, dass die Daten nicht durch Mehrfachzugriffe inkonsistent werden können. Greift ein weiterer Anwender auf die Daten zu, so muss dafür gesorgt werden, dass die aktuellen Daten bereitgestellt werden, oder der Datensatz/die Nachricht muss für andere gesperrt sein.

**Debugging und Tracing**

Für die Entwicklung der Systeme ist es in jeder Phase der Realisierung wichtig, dem Entwickler die Möglichkeit zu geben, seinen erstellten Code Schritt für Schritt zu kontrollieren. Diese Möglichkeit des Debuggings sollte die Entwicklungsumgebung des Integrationsservers bereitstellen. Dadurch kann der Entwickler nicht nur den Ablauf des erstellten Codes kontrollieren, sondern sich auch den Inhalt der Variablen zur Laufzeit anschauen. Durch die Erweiterungsprogrammiersprache des Integrationsservers kann man bereits vorab einschätzen, wie komfortabel das Debugging sein wird. Sind für die Programmierung allgemeine Programmiersprachen wie C/C++, JAVA, C# oder ähnliche vorgesehen, so ist die Wahrscheinlichkeit sehr hoch, dass ein komfortabler Debugger in der Entwicklungsumgebung bereits integriert ist oder zumindest der eines Drittanbieters verwendet werden kann. Handelt es sich dagegen um Skriptsprachen, muss man davon ausgehen, dass entweder kein oder nur ein Debugger mit geringem Funktionsumfang vorhanden ist. Im Gegensatz zum Debugging kann Tracing dafür verwendet werden, zur Laufzeit die Daten in eine Log-Datei zu schreiben. Hierbei ist zwischen einer Tracing-Möglichkeit, welche durch das Produkt angeboten wird, und einer Variante, welche der Entwickler in seinem Code einbaut, zu differenzieren. Das Produkt kann bereits die unterschiedlichsten internen Aktionen in eine

Log-Datei schreiben und dem Administrator und Entwickler diese Informationen zur Verfügung stellen. In dem durch den Entwickler erstellten Code muss während der Entwicklung festgelegt werden, welche Informationen in eine Log-Datei geschrieben werden sollen. Der Entwickler ruft in seinem Code entsprechende Funktionen auf, die die übergebenen Daten in die Log-Datei schreiben. Da diese Funktionsaufrufe zu Lasten der Performance gehen, ist darauf zu achten, dass der Umfang des Tracing über Parameter eingestellt werden kann. So kann man bei Bedarf das Tracing aktivieren und den Detailgrad der protokollierten Informationen bestimmen. Beim Debugging wie bei Tracing muss man sich darüber im Klaren sein, dass der Linker mehr Informationen in eine ausführbare Datei schreiben muss als ohne Debuggen/Tracing und so unter Umständen das System in seiner Gesamtgröße, bezogen auf den Speicherbedarf, anwächst. Außerdem wird durch Debuggen/Tracing das Laufzeitverhalten beeinflusst, was die Identifizierung von der Ausführungsgeschwindigkeit abhängiger Probleme erschwert. Trotz der genannten Nachteile ist das Debugging und das Tracing für einen effektiven Test notwendig und sollte immer durch den Integrationsserver unterstützt werden.

### **Skalierbarkeit**

Der Begriff Skalierbarkeit bedeutet, dass die Erweiterung der Hardware durch z.B. einen zusätzlichen Server oder Prozessor sich auch positiv auf die Performance des Gesamtsystems auswirkt. Grundsätzlich ist bei Leistungsproblemen vor einer Erweiterung durch zusätzliche Hardware eine Analyse durchzuführen, um jene Komponenten des Systems zu identifizieren, die den Flaschenhals darstellen.

### **Verteilbarkeit**

Unter Verteilbarkeit eines Integrationsservers versteht man die Aufteilung der Funktionalität des Integrationsservers auf mehrere Hardware-Server, ohne dass dies eine Veränderung der Anwendungen zur Folge hat. Die Applikationen benötigen also keine Anpassung der Kommunikation mit dem Integrationsserver, unabhängig davon, ob der Integrationsserver auf einem oder auf mehreren Hardware-Servern betrieben wird. Die Verteilung kann während der Installation durch den Administrator erfolgen, oder der Server führt dies in Abhängigkeit seiner Auslastung automatisch durch.

### **Weitere Eigenschaften**

Manche Eigenschaften eines Integrationsservers sind nur schwer vergleichbar. Neben den oben aufgezählten Punkten könnten auch noch die Begriffe Performance, Produktivität, Zuverlässigkeit, Qualität des Integrationsservers, Marktposition des Herstellers, unternehmensinterne Wartbarkeit und die Ausfallsicherheit genannt werden. In diesen Fällen kann man sich entweder auf die Herstellerangaben verlassen oder die Produkte durch eigene Tests miteinander vergleichen.

## **2.4 EAI-Architekturen**

Nach der allgemeinen Beschreibung von EAI folgt nun der technische Teil des Kapitels. Es werden die verbreitetsten Technologien dargestellt, die in EAI-Pro-

dukten zur Anwendung kommen. Die Gegenüberstellung der Technologien soll dem Leser helfen, die für ihn geeignetste Technologie zu finden. Eine Entscheidung für eine bestimmte Integrationsform und -art schränkt die Auswahl der Architektur stark ein.

### 2.4.1 Point-to-Point

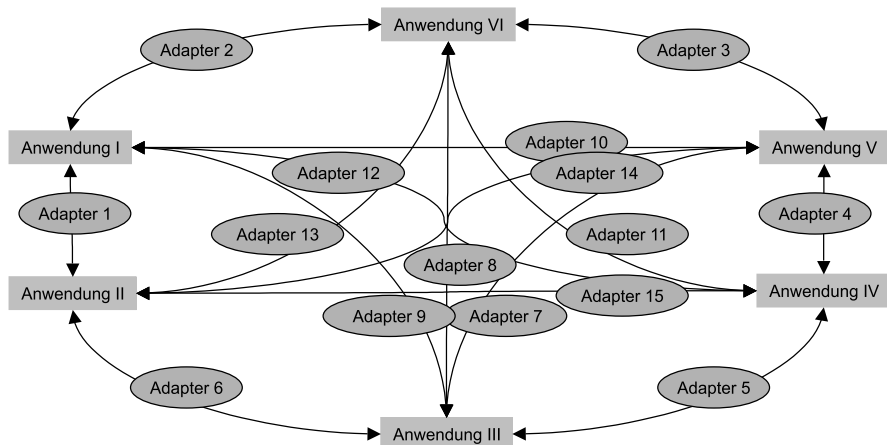
Mit einer Point-to-Point-Integration werden Applikationen direkt mit anderen Applikationen verbunden, die auf die jeweiligen Daten (ggf. sogar wechselseitig) zugreifen sollen. Dies bedeutet, dass für jede Verbindung zwischen den Anwendungen ein Adapter realisiert werden muss und die Abgrenzung zwischen Client und Server verwischt wird: jeder Rechner stellt einen Server und einen Client dar. Aus dieser Tatsache ergibt sich ein hoher Realisierungs- und Wartungsaufwand für die Schnittstellen. Dies kommt insbesondere dann zum Tragen, wenn eine Anwendung aus dem System gelöst werden soll oder durch eine neue Komponente ersetzt werden soll.

Vorteil:

- nicht bekannt

Nachteil:

- keine eindeutige Trennung zwischen Server und Client
- hoher Implementierungsaufwand
- hoher Wartungs- und Betriebsaufwand
- hohe Anzahl von Schnittstellen
- unüberschaubar und fehleranfällig



**Abbildung 2.14.** Point-to-Point

### 2.4.2 Hub & Spoke

Hierbei handelt es sich um eine Architektur, die von einem zentralen Hub (Verteiler) und um den Hub angeordneten Spokes (Speichen) ausgeht. Dies bedeutet, dass der Message Broker die zentrale Stelle für die Kommunikation der Applikationen untereinander darstellt. Die Kommunikation wird über die Adapter, die zwischen dem Message Broker (Hub) und der Anwendung implementiert werden, sichergestellt. Daraus ergeben sich bei  $n$  Applikationen nur  $n$  Schnittstellen und somit  $n$  Adapter. Als Produktbeispiel kann hier webMethods ([www.webmethods.com](http://www.webmethods.com)) genannt werden.

Vorteil:

- geringer Implementierungsaufwand
- flexible Architektur
- geringer Wartungs- und Betriebsaufwand
- geringe Anzahl von Schnittstellen

Nachteil:

- Leistungseinbußen durch den Server (Flaschenhals)

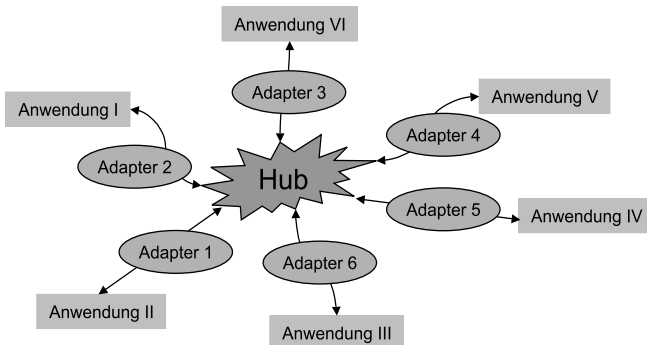


Abbildung 2.15. Hub & Spoke

### 2.4.3 Bus-oriented

Bei dieser Art der Architektur ist ein Bus die zentrale Verbindung zwischen den Applikationen. Über diesen Bus werden durch eine Master/Slave-Beziehung die Daten und Befehle zwischen den Anwendungen ausgetauscht. Die Anwendungen sind über Adapter an den Software-Bus angeschlossen, der durch den dezentralen Message Broker zur Verfügung gestellt wird. Die Anzahl der Adapter, die für die Anbindung an das IT-System realisiert werden müssen, ist somit wie bei Hub & Spoke gleich der Anzahl der Anwendungen, die an der Kommunikation beteiligt sind.



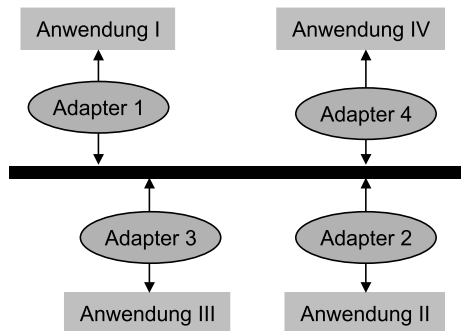
Mit einem Bus kann so das komplette IT-System mit einem geringen Aufwand und einer einfachen Struktur aufgebaut werden. Ein Produktbeispiel hierfür ist SeeBeyond ([www.seebeyond.com](http://www.seebeyond.com)).

Vorteil:

- bessere Skalierbarkeit
- gute Lastverteilung
- Auflösen von komplexen Strukturen

Nachteil:

- Falls kein EAI-Produkt verwendet wird, ist der Aufwand der Realisierung relativ hoch.



**Abbildung 2.16.** Bus-oriented

#### 2.4.4 Distributed Objects

Durch die Erweiterung des objektorientierten Konzeptes können Objekte im Netzwerk (dezentral) verteilt werden. Diese Objekte können autonom in einer großen Anzahl im Netzwerk auf unterschiedlichen Servern verteilt und über Adapter angesprochen werden. Die bekanntesten Technologien, die als Basis für diese Architekturart dienen, sind die Common Object Request Broker Architecture (CORBA) und Component Object Model/Distributed Component Object Model (COM/DCOM). Beide werden in den folgenden Kapiteln genauer beschrieben.

Vorteil:

- gute Lastverteilung

Nachteil:

- komplexe Programmierung

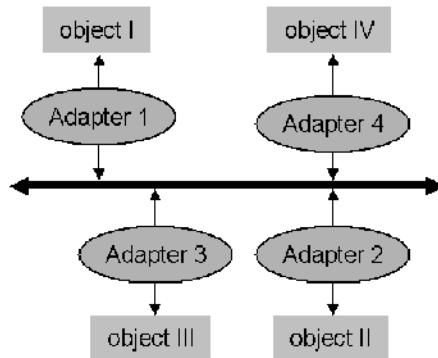


Abbildung 2.17. Distributed Objects

## 2.5 Middleware

Als letzte Komponente eines Integrationservers wird die Middleware beschrieben, die die grundlegende Infrastruktur für die Kommunikation zwischen verschiedenen Systemen zur Verfügung stellt. Die Middleware ist somit ein fester Bestandteil eines Integrationservers und kann durch die genannten Kommunikationsmodelle und -architekturen realisiert werden. Die klassische Middleware ist eine Sammlung von wenig spezialisierten Diensten, die zwischen der Systemplattform (der Hardware und dem Betriebssystem) und den Anwendungen angesiedelt sind und die Verteilung von Daten und Meldungen unterstützen. Eine Middleware ist außerdem plattformübergreifend und setzt auf den bereits genannten standardisierten Schnittstellen und Übertragungsprotokollen wie TCP/IP auf.

Da bisher keine einheitliche Definition veröffentlicht wurde, werden an dieser Stelle die gängigsten Kategorien genannt, aber nicht weiter beschrieben. Die verbreitetsten Kategorien sind:

### Datenbankorientierte Middleware

Es wird die Integration von Daten unterstützt, die auf verschiedenen Datenbanksystemen gespeichert sind. Die Datenbankhersteller bieten in der Regel Lösungen für ihr eigenes Produkt an, um ihr Datenbanksystem auf unterschiedliche Plattformen verteilen zu können.

### Funktionsorientierte Middleware

Es sind einzelne Module, die logisch zusammengehören, in aufrufbaren Funktionen zusammengefasst, um so eine erhöhte Wiederverwendbarkeit zu gewährleisten. Neben der Wiederverwendung dient der modulare Aufbau auch einer besseren Strukturierung der Anwendung. Der Aufruf von entfernten Funktionen ist unter dem Begriff Remote Procedure Call (RPC) bekannt.

**Transaktionsorientierte Middleware**

Eine Transaktion ist die Bearbeitung von einer oder mehreren miteinander verknüpften Funktionen, die anhand einer logischen Verarbeitungsfolge zusammengefasst sind. Diese Technologie wird seit längerem bei Datenbanken eingesetzt. Aus dem Datenbankbereich kommen auch die unterschiedlichen Eigenschaften von Transaktionen. Diese Eigenschaften sind unter dem Begriff ACID bekannt.

- Atomicity: Eine Transaktion wird immer vollständig oder gar nicht ausgeführt.
- Consistency: Die Transaktion führt von einem konsistenten Zustand zu einem anderen konsistenten Zustand.
- Isolation: Parallele Transaktionen werden so abgearbeitet, dass sie sich nicht beeinflussen.
- Durability: Das Ergebnis einer Transaktion ist beständig. Es überlebt das Sitzungsende und ist gegen Fehler (Software und Hardware) abgesichert.

**Messageorientierte Middleware**

Die Anwendungen tauschen Daten auf Basis von Nachrichten aus. Eine Nachricht besteht aus einer Datenfolge, die nur von der empfangenden Anwendung gelesen werden kann. Die Middleware stellt den Transport der Nachrichten zwischen den Anwendungen sicher und dient nicht als Anwendungsplattform wie transaktionsorientierte und komponentenorientierte Middleware.

**Komponentenorientierte Middleware**

Die komponentenorientierte Middleware ist eine objektorientierte Erweiterung der RPCs. Eine Komponente kann aus mehreren Objekten bestehen und aus weiteren Komponenten, die untereinander über eine synchrone Kommunikation Daten austauschen. Die bekanntesten Versionen einer komponentenorientierten Middleware sind CORBA und COM/DCOM (bzw. .NET).

**2.6 Kommentar**

Die hohe Anzahl der Integrationsmöglichkeiten und Technologien macht die Wahl der optimalen Anwendungsintegration nicht leichter. Für die Auswahl des richtigen Produktes und der richtigen Technologie kann auch keine generelle Empfehlung ausgesprochen werden, da die optimale Lösung von den genauen Anforderungen des Unternehmens abhängt. Es können aber einige Entscheidungskriterien für die Wahl der Technologien genannt werden. So kann zur Festlegung bezüglich der Kommunikationsart gesagt werden, dass die richtige Wahl der Technologie durch den Datenaustausch der einzelnen Datenobjekte bestimmt wird. Für alle Datenübertragungen, die keine sofortige Rückantwort benötigen, ist die asynchrone Kommunikationsart zu bevorzugen. Für alle Anfragen, deren Ergebnisse sofort für eine weitere Bearbeitung benötigt werden, stellt eine Kombination aus synchroner und den weiteren Kommunikationsarten die richtige Alternative dar. Beispielswei-

se werden Stammdaten asynchron aktualisiert und die Kontrolle von Lagerbeständen durch synchrone Abfragen realisiert.

Die oben genannte Technologieentscheidung kann mehr oder weniger unabhängig von einer zuvor durchgeführten Ist-Analyse getroffen werden. Voraussetzung für weitere darauf aufsetzende Festlegungen ist eine ausführlich durchgeführte Ist-Analyse bezüglich des technischen Ist-Zustandes des Unternehmens. So muss darauf geachtet werden, in welchem Dialekt die verwendeten Datenformate bisher tatsächlich verwendet werden. Kann dieser weiter verwendet werden oder ist es ein Dialekt, welcher sich nur schwer für die neuen Anwendungen verwenden lässt?

Geht man davon aus, dass man sich für eine prozessbasierte Integration entschieden hat, also zur primären Zielgruppe dieses Buches gehört, so wurde die Entscheidung bezüglich der Architektur bereits eingeschränkt und man benötigt zur Realisierung entweder eine EAI-Plattform, in der ein WfMS vorhanden ist, oder ein WfMS mit der Möglichkeit, bereits bestehende Anwendungen über Adaptertechnologie anbinden zu können. An dieser Stelle wird auf die Kapitel Softwarearchitektur einer workflow-basierten Lösung und Software-Architekturen verwiesen.

Neben den Technologieanforderungen sind die Kosten für ein solches Projekt nicht ganz unerheblich. So sollte man davon ausgehen, dass nicht die EAI-Produktlizenzen die Kostentreiber sind, sondern die Adapter, die realisiert werden müssen, um die (bestehenden) Anwendungen anzubinden. Die Erfahrung hat gezeigt, dass eine Kostenverteilung 20/80 vorliegt: 20 % der Kosten müssen für die Lizenzen der EAI-Plattform gerechnet werden, und 80 % für die Anbindung der vorhandenen Applikationen. Dies kann natürlich nur als Richtwert dienen. Es wird aber damit ausgedrückt, dass die Adapter der Teil der Lösung sind, mit dem die Kosten positiv durch die entsprechende Architektur beeinflusst werden können. Plant man z.B. standardisierte und normalisierte Schnittstellen, so können die Kosten für die Realisierung und die Wartung gesenkt werden. Dies macht sich dann besonders bemerkbar, wenn die Lösung durch weitere Funktionalitäten/Komponenten ergänzt werden soll. Durch die Verwendung von standardisierten Schnittstellen ist die Wahrscheinlichkeit sehr hoch, eine passende Anwendung zu finden, die die geforderte Funktionalität abdeckt.

Ist eine Analyse Ihres IT-Umfeldes durchgeführt worden und wurden grundlegende Entscheidungen bezüglich der Architektur und Technologien getroffen, so kann für eine Produktauswahl eine Einstufung der EAI-Produkthersteller der *Gartner Group* herangezogen werden. Darin werden die Produkthersteller in Marktführer, Visionäre, Herausforderer und Nischenanbieter eingestuft.

### **Marktführer**

bieten eine fast komplette Entwicklungsumgebung an, um Daten unter Berücksichtigung von Prozesssteuerungen umzuformen und weiterzuleiten. Adapter für wichtige Software-Produkte sind vorhanden. Auch die notwendige Unterstützung

zum sinnvollen Einsatz der Entwicklungsumgebung in Ihrem Unternehmen können diese Unternehmen bieten.

**Visionäre**

Sie zeichnen sich durch durchgängige und interessante Konzepte aus. Diese Konzepte sind aber nur zum Teil realisiert.

**Herausforderer**

Sind Unternehmen, die über das nötige Geld und die Marktmacht verfügen, um die Produkte in ihrem Portfolio weiter auszubauen, dies aber abgesehen von Marketingaktionen noch nicht getan haben.

**Nischenanbieter**

Sind Unternehmen, die ihre Produkte nicht für die Masse entwickelt haben, sondern für spezielle Aufgaben optimieren.

**Tabelle 2.1.** Produktkategorien

Kategorie	Produkte bzw. Unternehmen
Marktführer	IBM, See Beyond, Tibco, Vitria, webMethods.
Visionäre	Bea, Crossworlds, Iona, iPlanet, Mercator, Sybase
Herausforderer	Microsoft, Oracle
Nischenanbieter	Candle, Fujitsu Siemens, WRQ

Nachdem es am Anfang der „EAI-Bewegung“ den Produktanbietern primär um den Datenfluss zwischen den Applikationen ging, hat sich in den letzten Jahren ein Wandel vollzogen. Dies hat zur Folge, dass die großen EAI-Anbieter entweder Kooperationen mit WfMS-Herstellern eingegangen sind, oder die WfMS-Hersteller von den EAI-Anbietern gekauft wurden. Somit ist der Bedarf und die Notwendigkeit, die Integration durch eine Prozesssteuerung zu unterstützen, erkannt und abgedeckt worden. Im Folgenden Kapitel werden die gängigsten Software-Architekturen beschrieben, um die Basis für die prozessorientierte Integration darzustellen. Die prozessorientierte Integration selbst wird dann im Kapitel Workflow-based Integration (WfBI) beschrieben.

### 3 Software-Architekturen

Um eine IT-Lösung entwickeln zu können, stehen verschiedene Ansätze bezüglich der Software-Architektur zur Verfügung. Der Leser wird in diesem Kapitel einen Überblick gewinnen, welche objektorientierten Architekturen im Zusammenhang mit WfMS- und EAI-Lösungen möglich sind. Als erstes wird aber erklärt, wie eine Software-Architektur definiert ist, welche Bestandteile sie kennzeichnen und wozu eine Software-Architektur dient. Zu diesem Zweck wird nachfolgend die Definition nach Bass, Clements und Kazman zitiert.

„The software architecture of a program or computing system is the structure or the structures of the system, which comprise software components, the externally visible properties of those components, and the relationships among them.“ (Bass et al. 1998).

Die in dieser Definition genannten Begriffe Komponenten, Eigenschaften und Beziehungen sollen näher erklärt werden. Die Komponenten stellen eine Gruppierung dar, in der alle Funktionen einer bestimmten fachlichen Anforderung gekapselt sind. Als Beispiel für eine Komponente kann eine Datenbank, ein Server, ein Client oder ein Objekt genannt werden. Durch die Beziehungen der Komponenten untereinander und die Eigenschaften dieser Beziehungen entsteht eine Struktur, die man Software-Architektur nennt. Eine solche Struktur beschreibt die funktionalen und qualitativen Merkmale des IT-Systems und dient als Diskussionsgrundlage für weitere Entscheidungen über das Design und die Realisierung des IT-Systems. Die Voraussetzung dafür ist eine sorgfältige Dokumentation der Software-Architektur. Eine solche Dokumentation muss von jedem Projektbeteiligten, von der Geschäftsführung bis hin zum Sachbearbeiter und Entwickler, gelesen und verstanden werden können. Diesen Anspruch deckt die Unified Modeling Language (UML) mit ihren Diagrammen ab. Da in der UML-Notation eine Vielzahl von unterschiedlichen Diagrammen angeboten wird, werden für die Dokumentation der Software-Architektur fünf dieser Diagramme verwendet (Anwendungs-, Klassen-, Paket-, Interaktions- und Implementierungsdiagramme). Es soll auch darauf hingewiesen werden, dass dies nicht die einzige mögliche Art von Diagrammen bzw. Sichten auf eine Software-Architektur ist. Als weitere Beispiele können die vier Sichten von Hofmeister, Nord und Soni und das FMC Fundamental Modeling Concept für die Darstellung einer Software-Architektur genannt werden.

Die UML-Notation ist nach Meinung des Autors die am besten geeignete Dokumentationsform, da sie nicht nur die Analyse, das Design und die Implementierung eines IT-Projekts begleitend dokumentieren kann, sondern auch die Architek-

tur von Workflow-Lösungen unterstützt, da ab der UML 2.0 ein Prozessdiagramm enthalten ist. In Tabelle 3.1 werden jene Diagrammtypen kurz beschrieben, die bei der Realisierung von Workflow-Projekten, also für die Umsetzung der IT-Lösung, genutzt werden können.

Im Zusammenhang mit der Prozessmodellierung wird die UML in dem Kapitel Unified Modeling Language (UML) detaillierter beschrieben.

**Tabelle 3.1.** UML-Diagramme

Diagrammname	Beschreibung
Anwendungsfalldiagramm (Use Case Diagram)	Beschreibung der Geschäftsprozesse, welche in dem System realisiert werden sollen
Klassendiagramm (Class Diagram)	Beschreibung der Klassen mit ihren Abhängigkeiten (statische Struktur des Systems)
Paketdiagramm (Package Diagram)	Gruppierung der Klassen in fachlichen Modulen
Interaktionsdiagramm	Kann unterschieden werden in Sequenzdiagramm (klassenorientierte Aufrufstruktur), welches die zeitliche Abfolge beschreibt, und Kollaborationsdiagramm (nachrichtenorientierte Aufrufstruktur), welches den Nachrichtenfluss (Datenfluss) zwischen den Objekten beschreibt.
Zustandsdiagramm (State Chart Diagram)	Stellt das dynamische Verhalten des Systems dar. Es wird das Verhalten der Objekte beschrieben.
Aktivitätsdiagramm (Activity Diagram)	Beschreibt die Abläufe (Kontrollfluss) des Prozesses.
Implementierungsdiagramm	Es wird die (verteilte) Anwendung und die dazu notwendige Hardware beschrieben. Man kann hier unterscheiden zwischen dem Komponentendiagramm, welches die Zusammenhänge der Software-Komponenten beschreibt, und dem Deployment-Diagramm, welches die Hardware-Struktur beschreibt.

Sind die sieben Diagramme erstellt, liegt die Dokumentation der Software- und Hardware-Architektur vor, in der die IT-Lösung mit allen Komponenten beschrieben ist. Durch die unterschiedlichen Darstellungen können alle – vom Anwender bis zum Administrator, der den späteren Betrieb sicherstellen muss – diese Dokumentation lesen.

Zu Beginn des Kapitels wurden grundlegende Merkmale bezüglich der Software-Architekturen und eine Möglichkeit einer Dokumentation vorgestellt. Bevor Im

Folgenden zwei Architekturarten zur Realisierung von WfMS- und EAI-basierten Lösungen im Detail vorgestellt werden, soll auf die Anforderungen oder auch Qualitätsmerkmale einer Software-Architektur eingegangen werden. Es ist erforderlich, zwischen zwei qualitativen Eigenschaften der Anforderungen zu unterscheiden, welche durch Bass (Bass et al. 1998) wie folgt beschrieben werden:

1. Anforderungen, die bei dem Verwenden der IT-Lösung erkennbar sind. Darunter fallen die folgenden Kategorien: Funktionalität, Benutzerfreundlichkeit, Performance, Sicherheit, Betriebssicherheit und Zuverlässigkeit.
2. Anforderungen, die bei der Realisierung und bei der Wartung der IT-Lösung beobachtet werden. Darunter fallen die folgenden Kategorien: Wartbarkeit, Wiederverwendbarkeit, Integrierbarkeit und Testbarkeit.

Um den Anforderungen einer modernen Software-Architektur gerecht zu werden, müssen die folgenden Qualitätsmerkmale beachtet werden.

### **Modularität**

Die Modularität eines IT-Systems ist dann erfüllt, wenn fachliche Funktionen in Module gekapselt sind und diese Module individuell verknüpft werden können. Als Beispiel kann man eine Office-Anwendung nennen, in der Textverarbeitung, Mail-Client, Tabellenkalkulation und Präsentationsprogramm enthalten sind. Diese einzelnen Programme (Module) können unabhängig voneinander, aber auch zusammen auf der gleichen Programmbasis verwendet werden; somit ist eine Wiederverwendung der Module erreicht worden. Den Grad der Modularität kann man beliebig fein gestalten. In dem eben genannten Beispiel befindet man sich auf Programmebene. Ein feiner Grad wäre die Modulebene: z.B. die Gruppierung aller Datei-Funktionen und aller Ansichts-Funktionen in jeweils einem Modul. Wird die Modularität konsequent in der Architektur berücksichtigt, so hat dies auch einen positiven Einfluss auf die Wartbarkeit des IT-Systems.

### **Funktionalität**

Der Begriff Funktionalität beschreibt die Anforderungen an eine IT-Lösung oder auch an einzelne Module. Dieselbe Funktionalität kann unterschiedliche Architekturen als Realisierungsbasis haben.

### **Erweiterbarkeit**

Die gute Erweiterbarkeit einer IT-Lösung bzw. einer Architektur ist dann gegeben, wenn Module in ihrer Funktionalität erweitert werden können, ohne dass andere Module aufgrund dieser Erweiterung angepasst werden müssen.

### **Skalierbarkeit**

Die Skalierbarkeit ist für große IT-Systeme wichtig, um z.B. eine wachsende Anwenderzahl performant bedienen zu können. Dies kann dadurch erreicht werden, dass Module auf eine zusätzliche Hardware, z.B. in Form eines weiteren Servers, verlegt werden. Dadurch wird die Performance des IT-Systems erhöht.



**Performance**

Unter der Performance einer IT-Lösung versteht man die zeitliche Leistung der Funktionalität. Die Performance kann durch Messung der Reaktions- und Bearbeitungszeit der Funktion sowie die Anzahl an Transaktionen pro Zeiteinheit bestimmt werden.

**Stabilität und Verfügbarkeit**

Die Stabilität und die Verfügbarkeit von IT-Systemen werden mit der Ausfallzeit des Systems bewertet. Die Ausfallzeit setzt sich aus der Reaktionszeit des Supports und dem zeitlichen Aufwand für die Reparatur des Systems zusammen. Eine Optimierung der Ausfallzeit kann durch das Bereitstellen von redundanten Systemen oder auch nur Teilen davon erreicht werden. In diesem Fall ist es wichtig, die Architektur entsprechend darauf auszurichten, d.h. jene Module, welche eine hohe Verfügbarkeit haben sollen, müssen in dem modularen Aufbau entsprechend berücksichtigt werden und z.B. in einer redundanten Ausführung implementiert werden.

**Sicherheit**

Unter der Sicherheit eines IT-Systems versteht man in diesem Fall nicht die Fallsicherheit, sondern den Schutz der Daten und Funktionen des IT-Systems vor unberechtigtem Zugriff. Sind Zugänge von außen (z.B. über das Internet) an das IT-System vorgesehen, ist auch darauf zu achten, dass diese ausreichend gegen mögliche Angreifer geschützt sind. Zusätzlich ist die Gefahr von innen, also durch eigene Mitarbeiter, nicht zu unterschätzen und bedarf einer detaillierten Betrachtung.

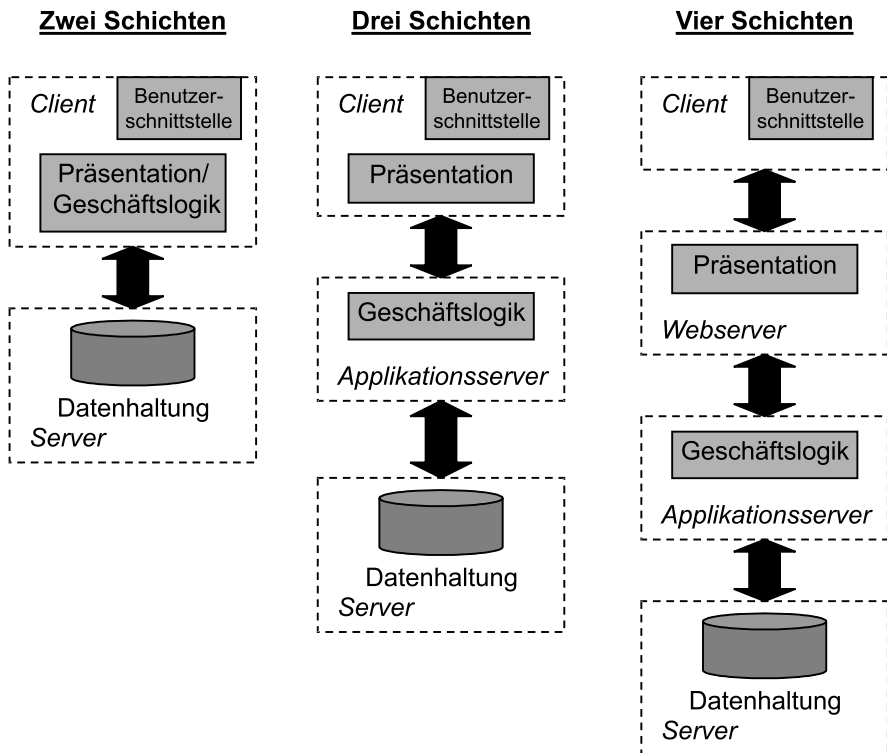
Es wird darauf hingewiesen, dass allen Qualitätsmerkmalen die gleiche Bedeutung zuteil werden muss, um eine hohe Investitionssicherheit zu erreichen. Ein Qualitätsmerkmal, auf das ein besonderes Augenmerk gerichtet werden muss, ist die Sicherheit einer Architektur. Diese Anforderung ist für ein WfMS besonders im Web-Umfeld von sehr großer Bedeutung, da durch gezielte Angriffe von außen in die Geschäftsprozesse eingriffen werden kann bzw. die Gefahr besteht, dass Geschäftsprozesse oder deren Daten ausspioniert werden.

Bevor eine Client/Server- und eine webbasierte Software-Architektur beschrieben werden, soll noch auf das Schichtenmodell eingegangen werden. Die Architekturen können in verschiedene Schichten aufgeteilt werden, um eine klare Trennung bezüglich der Funktionalität und Aufgaben zu erreichen. Es werden z.Zt. zwischen zwei und vier Schichten in einem WfMS realisiert. Es können auch mehr als vier Schichten realisiert werden. In einem solchen Fall werden einzelne Schichten (Benutzerschnittstelle, Präsentation, Geschäftslogik und Datenhaltung) weiter unterteilt. Zwischen den Schichten sind unterschiedliche Merkmale der Trennung möglich. In Anlehnung an das ISO-Schichtenmodell können nur benachbarte Schichten direkt miteinander kommunizieren, d.h. eine Schicht kennt nur die direkt über ihr liegende Schicht und die direkt darunter liegende Schicht. Der Realisierungs- und Wartungsaufwand ist in diesem Fall geringer als bei einer schichtübergreifenden Kommunikation. Die zuerst genannte Art der Trennung bezeichnet

man als eine harte Trennung der Schichten. Dagegen spricht man von einer weichen Trennung, wenn eine schichtenübergreifende Kommunikation realisiert wird. Diese kann relativ schnell unübersichtlich und wartungsanfällig werden: die Kommunikationswege laufen nicht kanalisiert über die nächstliegende Schicht! Das kann zur Folge haben, dass bei Änderungen in einer Schicht eventuell alle verwendeten Schichten angepasst werden müssen und nicht wie bei einer harten Trennung nur die darüber oder darunter liegende Schicht.

Es sei noch darauf hingewiesen, dass die Anzahl der Schichten keinen unbedingten Einfluss auf die Hardware-Architektur hat. So ist es z.B. möglich, eine Drei-Schichten-Architektur auf einem einzigen Server zu realisieren. Es ist aber auch möglich, die beiden untersten Schichten auf zwei unterschiedlichen Servern zu implementieren.

In dem folgenden Abschnitt werden die unterschiedlichen Schichtenmodelle einer Software-Architektur beschrieben.



**Abbildung 3.1.** Software-Architekturarten

### 3.1 Software-Architekturarten

#### 3.1.1 Zwei-Schichten-Architektur

Eine Zwei-Schichten-Architektur ist eine typische Client/Server-Lösung, in der der Server die Datenbank beherbergt und die Zugriffe auf die Daten über den Client realisiert werden. Der Client beherbergt in diesem Modell die Präsentations- und die Geschäftslogik. Diese sind gemeinsam auf einen Rechner installiert, damit die Daten auf dem Server bearbeitet werden können.

Vorteil:

- Die Lösungen sind ohne großen Aufwand zu realisieren.
- Multiuser Handling wird durch die Datenbank verwaltet.

Nachteil:

- Eine Änderung am Client hat in der Regel eine Neuinstallation der Client-Software zur Folge.
- Jede Datenbankverbindung geht zu Lasten der Performance des Datenbankservers und des Netzwerkes.
- Jeder Client muss dasselbe Betriebssystem haben, es sei denn, der Anwendungs-Client ist in einer betriebssystemunabhängigen Programmiersprache (Java) realisiert.
- Eine „Programmierdisziplin“ ist nur schwer einzuhalten, da der Programmierer nicht gezwungen wird, die Präsentations- und Geschäftslogik zu trennen.

#### 3.1.2 Drei-Schichten-Architektur

In einer Drei-Schichten-Architektur wird die Präsentationslogik von der Geschäftslogik getrennt. Dies ermöglicht, dass der Client nur die Präsentationslogik enthält und somit „schlanker“ wird. Die dritte Schicht stellt eine auf einem Applikationsserver installierte Geschäftslogik dar, die den Datenbank-Server mit den Clients verbindet.

Vorteil:

- Der Grad der Wiederverwendung erhöht sich durch die Trennung der Präsentationslogik von der Geschäftslogik. Dadurch kann die Geschäftslogik in weiteren Anwendungen verwendet werden.
- Die Skalierbarkeit der IT-Lösung verbessert sich, da bei steigendem Bedarf an Performance die Geschäftslogik auf weitere Applikationsserver verteilt werden kann.
- Die Anzahl der Datenbankverbindungen kann minimal bis auf eine begrenzt werden; dies hat einen Performancegewinn zur Folge.
- Die Client-Lizenzen für die Datenbank können reduziert werden.

Nachteil:

- Eine aufwändige Realisierung des Applikationsservers durch einen Multiuser- und Multithreading-Mechanismus. Dies kann durch den Kauf oder die Entwicklung eines passenden Frameworks verringert werden.
- Die Architektur lässt keine browserbasierten Lösungen zu, und somit kann nur ein eigens entwickelter Client über das Inter-/Intranet mit dem Applikationsserver kommunizieren.

### 3.1.3 Vier-Schichten-Architektur

Die Veränderung gegenüber der Drei-Schichten-Architektur ist die Verlagerung der Präsentationslogik vom Client zum Applikationsserver. Somit bleibt auf dem Client nur noch die Benutzerschnittstelle, welche nur die Visualisierung der Daten durchführt und die eingegebenen Daten an den Server weitergibt. Mit einer Vier-Schichten-Architektur können browserbasierte Anwendungen für das Inter- und Intranet realisiert werden. Zu diesem Zweck wird die Präsentationslogik in einen Webserver und die Geschäftslogik in einen Applikationsserver verlagert. Die beiden genannten Server sind Programme, die diese Funktionalitäten zur Verfügung stellen. Sie können bei Bedarf auf zwei getrennten HW-Servern betrieben werden.

Vorteil:

- Änderungen in der Präsentationslogik und in der Geschäftslogik können unabhängig voneinander durchgeführt werden.
- Die Präsentationslogik einer weiteren Anwendung kann durch eine Trennung von Präsentations- und Geschäftslogik dieselbe Geschäftslogik wieder verwenden.
- Der Client wird in der Regel in einem Browser betrieben und ist somit betriebs-systemunabhängig und muss nur einmal installiert werden.
- Der Client ist ein Thin-Client und benötigt somit wenig HW-Ressourcen.

Nachteil:

- Eine aufwändige Realisierung des Applikationsservers durch ein Multiuser- und Multithreading-Verfahren. Dies kann durch den Kauf oder die Entwicklung eines passenden Frameworks verringert werden.
- Wird der Client in einem Browser betrieben, so bleibt die Funktionalität gegenüber einem konventionellen Client sehr eingeschränkt.

Nach den allgemeinen Erklärungen zu einer Software-Architektur wird nun auf spezielle Architekturen eingegangen.

## 3.2 Java 2 Enterprise Edition (J2EE)

Um serverseitige Software-Komponenten zu entwickeln, die über das Internet oder Intranet zur Verfügung gestellt werden sollen, ist die Verwendung einer J2EE-Architektur empfehlenswert. J2EE ist nicht nur eine Architektur, sondern auch eine Spezifikation, die es ermöglicht, Komponenten von unterschiedlichen Herstellern miteinander zu verknüpfen. Um die Möglichkeit zu schaffen, möglichst viele fachliche und technische Anforderungen abdecken zu können, wurden in der Spezifikation verschiedene Container und vier verschiedene Schichten in unterschiedlichen Modellen definiert. Die Kommunikation zwischen den Containern, den Schichten und zu anderen Komponenten ist ebenso in der J2EE-Spezifikation definiert. Zunächst werden die einzelnen Schichten (Layer) und ihre Eigenschaften beschrieben.

### Data Layer

Der Data Layer enthält das Enterprise-Information-System (EIS), welches in der Regel durch eine relationale Datenbank realisiert wird. Diese ermöglicht es, die Daten permanent, performant und sicher zu speichern. Es besteht aber auch die Möglichkeit, über einen Mainframe oder einen Message-Server den Zugriff auf die Daten zu erhalten. Die Art der Kommunikation mit den darüber liegenden Schichten wird durch die Verwendung der Datenquelle bestimmt. Wird eine relationale Datenbank verwendet, wird diese über einen datenbankunabhängigen JDBC-Konnektor oder (im Falle eines Message-Servers) über den Java Message Service (JMS) mit der Applikation verbunden.

### Business Layer

Im Business Layer liegt die Geschäftslogik der Anwendung. Sie wird meistens mit Enterprise Java Beans (EJB) realisiert. Die EJBs werden in dem EJB-Container installiert und über den Container den Anwendungen zur Verfügung gestellt. Die Methoden der EJBs werden über die Protokolle RMI-IIOP oder RMI-JRMP aus dem Web-Layer aufgerufen. Um eine Anwendung sicher zu gestalten, werden die EJBs nie direkt aus dem Internet aufgerufen, sondern die Kommunikation wird immer über einen Web-Layer nach außen sichergestellt. Technisch sollte ein direkter Aufruf durch eine Firewall verhindert werden.

### Web Layer

Der Web Layer enthält die Präsentationslogik, mit der die Geschäftslogik der Anwendung im Internet und/oder im Intranet dargestellt wird. Dies wird durch die Web-Komponententechnologie JavaServlets und JavaServer Pages (JSP) realisiert. Servlets und Java Server Pages werden über http aufgerufen und liefern als Antwort dieser Anfrage ein HTML-Dokument, ggf. zusammen mit einer Ergebnismenge, zurück. Die Ergebnismenge kann Daten aus dem EIS enthalten.

### Client Layer

Der Client kann in dem Client Layer als Web-Browser zur Darstellung von HTML-Dokumenten oder als Java-Anwendung realisiert werden. Grundsätzlich ist der Client-Layer als Eingabemöglichkeit und Darstellungsmöglichkeit von Daten definiert.

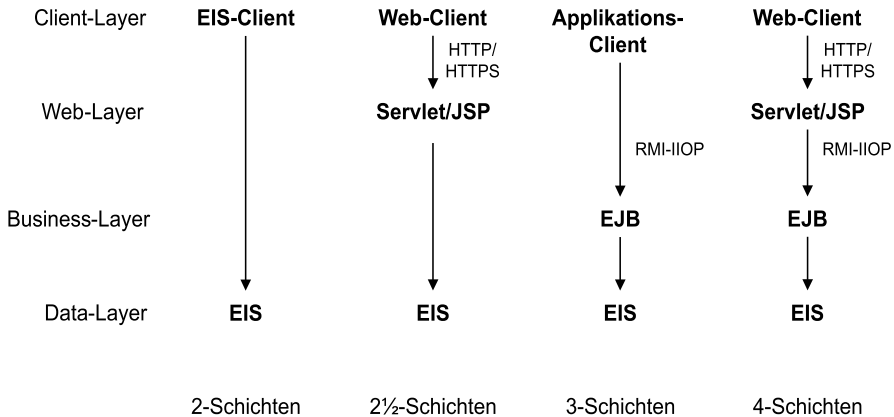


Abbildung 3.2. J2EE-Schichten

Im Folgenden werden die Architekturen beschrieben, die mit den zuvor beschriebenen 2-, 3- und 4-Schichten-Modellen weitgehend übereinstimmen. In der J2EE-Architektur werden diese Modelle noch um ein 2½-Schichtenmodell ergänzt.

In der Zwei-Schichten-Architektur wird der Client als EIS-Client bezeichnet, da er die Präsentations- und Geschäftslogik der Anwendung enthält und die Daten direkt aus der Datenbank ausliest. Für die Kommunikation wird beispielsweise ein JDBC-Treiber verwendet.

Um eine Trennung zwischen Präsentations- und Geschäftslogik zu erreichen, wird die 3-Schichten-Architektur verwendet. Hier wird die Anwendung im Client-Layer als Applikations-Client bezeichnet. Der Client enthält die Präsentationslogik und ruft die Methoden der EJBs auf, die die Geschäftslogik im Business-Layer enthalten. Diese Architektur findet, wegen des oben beschriebenen Sicherheitsrisikos, hauptsächlich in einem Intranet Verwendung. Soll eine Anwendung im Internet zur Verfügung gestellt werden, muss eine 2½- oder 4-Schichten-Architektur realisiert werden. Hierfür wurde der Web-Layer definiert: er stellt die Web-Komponenten zur Verfügung, welche die Anfragen über HTTP/HTTPS bedienen. Der wichtigste Unterschied zwischen den beiden Internet-tauglichen Architekturen ist, dass in der 2½-Schichten-Architektur die Präsentations- und die Geschäftslogik auf einem Server realisiert sind, aber die Geschäftslogik nicht von anderen Anwendungen verwendet werden kann. Deshalb muss diese Architektur eigentlich als eine 2-Schichten-Architektur angesehen werden. Dieser Nachteil wurde in der 4-Schichten-Architektur beseitigt. Mit ihr wird die Präsentationslogik in einem Web-Layer realisiert und kommuniziert mit der im Business-Layer enthaltenen Geschäftslogik. Somit kann die Geschäftslogik von anderen Anwendungen genutzt und die Anwendung über einen Web-Client im Internet zur Verfügung gestellt werden. Dieser Web-Client wird in der Regel in einem Browser betrieben. Es besteht aber auch die Möglichkeit, dass eine Java-Anwendung direkt die Methoden der EJBs aufruft.

Neben den oben beschriebenen Schichten sind die technischen Module in der J2EE-Spezifikation definiert. Darin werden Komponenten unterschieden, die Bestandteile von Anwendungen sind. Im Mittelpunkt stehen aber die Container, welche die Komponenten verwalten und die Dienste bereitstellen, mit denen die Komponenten arbeiten. In der J2EE-Spezifikation sind vier Container mit unterschiedlichen Aufgaben definiert. Diese Container findet man sowohl auf dem Server (Web-Container und EJB-Container) als auch auf dem Client (Applet-Container und App-Client-Container). Dies ermöglicht eine definierte Kommunikation zwischen den Containern anhand unterschiedlicher Protokolle. Wie aus Abbildung 3.3 hervorgeht, kann der Client-Container mit dem Web-Container über HTTP/HTTPS kommunizieren. Zusätzlich ermöglicht es der App-Client-Container, über RMI/IIOP direkt EJBs in dem EJB-Container aufzurufen. Über RMI kommuniziert auch der Web-Container mit dem EJB-Container.

Die verbreiteste Client-Umgebung eines J2EE-Client ist der Web-Browser, der generierte oder programmierte HTML-Seiten darstellt. Enthalten die HTML-Seiten Applets, dann verhält er sich als Applet-Container, stellt aber sonst keine weiteren Dienste zur Verfügung. Ein Applikations-Client-Container ist dagegen ein in Java geschriebener Client, der neben den Möglichkeiten, mit dem Web- und EJB-Container Daten auszutauschen, auch die Möglichkeit bietet, über bereitgestellte Dienste mit weiteren Komponenten wie z.B. einer Datenbank zu kommunizieren. Der Web-Container enthält zwei Arten von Web-Komponenten, die als Benutzeroberfläche einer Web-Anwendung genutzt werden können: Die Benutzeroberfläche kann durch Servlets und/oder Java Server Pages realisiert werden, die sich in der Art der Funktionalität unterscheiden. Ein Servlet hat die Möglichkeit, die Dienste des Web-Containers zu verwenden, während eine Java Server Page HTML oder XML mit eingebettetem Code enthält, der einen dynamischen Inhalt beschreibt. Dieser Code wird von einer JSP-Engine interpretiert und erstellt ein ausführbares Servlet.

Ein EJB-Container stellt ähnliche Dienste wie der Web-Container bereit, enthält aber die Enterprise Java Beans (EJB), die die Anwendungsfunktionalität, Geschäftsobjekte und die Geschäftslogik enthalten. Die EJBs werden in drei Arten (Session-, Entity- und Message Driven Bean) zur Verfügung gestellt und unterscheiden sich in ihrem Lebenszyklus, ihrem Verwendungszweck und in der Art, wie sie aufgerufen werden.

### **Session Bean**

Eine Session Bean ist nicht persistent, d.h. sie besteht nicht über die Dauer einer einzelnen Sitzung hinaus. Sie bezieht sich immer nur auf einen Client und kann den Client-spezifischen Status stateless oder stateful annehmen.

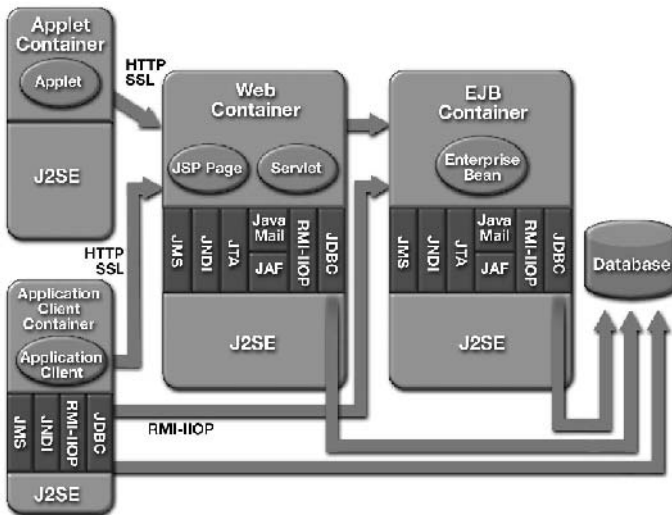
### **Entity Bean**

Eine Entity Bean dagegen ist langlebig und repräsentiert die Daten eines fachlichen Objektes, das in einer Datenbank bzw. einem EIS gespeichert ist. Sie kann gleichzeitige Datenzugriffe mehrerer Clients bearbeiten und unterliegt der Transaktionsverwaltung.

### Message Driven Bean (MDB)

Ab der EJB-Version 2.0 steht die MDB zur Verfügung. Sie ist eine Erweiterung der oben beschriebenen Beans und unterscheidet sich hauptsächlich in der Art der Kommunikation. Die MDB kommuniziert nicht wie die anderen beiden Bean-Arten synchron mit ihren „Nachbarn“, sondern asynchron über Nachrichten. Eine MDB enthält somit die Geschäftslogik für die Verarbeitung asynchroner Nachrichten.

In der J2EE-Spezifikation sind eine Vielzahl von Diensten, Kommunikationsarten und Messaging-Standards enthalten. Für weiterführende Informationen sei an dieser Stelle auf das J2EE-Tutorial von Sun Microsystems verwiesen.



**Abbildung 3.3.** J2EE-Komponenten (Darstellung von Sun Microsystems)

Neben der Architektur werden in der J2EE-Spezifikation auch die Rollen und das dazugehörige Know-how der beteiligten Entwickler über den gesamten Lebenszyklus einer Anwendung definiert.

### J2EE Product Provider

Er ermöglicht eine J2EE-Plattform mit Container, Java APIs, Werkzeugen usw. und ist ein Anbieter von Applikationsservern, Datenbanksystemen oder Web-Servern.

### Tool Provider

Er ist ein Anbieter von Werkzeugen für das Entwickeln von Anwendungen. Darunter zählen Werkzeuge für das Bereitstellen (Deployment) und die Verpackungen (Packaging), die Überwachung (Monitoring) und die Administration der Anwendungen.



**Application Component Provider**

Er entwickelt Web-Komponenten, Enterprise Beans, Applets oder Applikations-Clients, welche in Anwendungen verwendet werden können. Diese Rolle unterteilt sich in die genannten Komponenten.

**Application Assembler**

Er erhält die Applikationskomponenten durch den Application Component Provider in Form von JAR-Dateien<sup>3</sup> und baut diese gemäß der fachlichen Anforderungen zu einer J2EE-Applikation zusammen.

**Application Deployer and Administrator**

Sie bringen die Anwendung in einen konkreten Kontext durch die Konfiguration der Komponenten und die Auflösung der durch den Application Assembler definierten Abhängigkeiten. Zu seinen Tätigkeiten gehört z.B. die Abbildung (Mapping) auf eine bestimmte Datenbank.

Je nach Projektgröße müssen die oben beschriebenen Rollen nicht durch jeweils eine einzelne Person besetzt werden. Während in Großprojekten mehrere Personen eine Rolle besetzen können, ist es in kleineren Projekten auch möglich, dass eine Person mehrere Rollen inne hat.

### 3.3 Weitere objektorientierte Architekturen

Im Folgenden werden noch weitere Architekturen für das Erstellen von verteilten Anwendungen beschrieben. Verteilte Anwendungen sind nicht erst in jüngster Vergangenheit ein Trend in der Software-Entwicklung. Der Wunsch, über ein Netz verteilte, objektorientierte Anwendungen zu erstellen, kam bereits Ende der 80er Jahre auf. Der Aufwand für eine solche Anwendung war aber aufgrund der direkten Verwendung von Kommunikationsprotokollen sehr hoch. Aufgrund fehlender Standards konnten die Anwendungen nur sehr systemnah und nur durch wenige Spezialisten erstellt werden. Auch war an eine Verwendung in einer heterogenen Systemlandschaft mit unterschiedlichen Betriebssystemen und Programmiersprachen nicht zu denken. Mit der Objektorientierung wurde eine Basis für Mechanismen wie Datenkapselung durch Klassen geschaffen, die das Verbergen der Details des zugrunde liegenden Betriebssystems und des verwendeten Kommunikationsprotokolls zulassen. In den Konzepten wurden Objekte als Verteilungseinheiten verwendet, auf die durch Proxies ein transparenter entfernter Zugriff ermöglicht wurde. Für den Benutzer bleibt durch die Einführung eines zentralen Nachrichtenbrokers verborgen, an welchem Ort sich das aufgerufene Objekt physikalisch befindet. Dies war der Grundstein für das 1989 gegründete herstellerunabhängige Konsortium OMG (Object Management Group, [www.omg.org](http://www.omg.org)). Die OMG hat 1989 die Spezifikation für die Common Object Request Broker Architecture (CORBA) erstellt und verabschiedet. Sie achtet darauf, dass der Standard sich im-

---

<sup>3</sup> JAR-Dateien (Java ARchive) sind Archive, die es ermöglichen Java-Module in einer Datei zu archivieren.

mer an die neuen Anforderungen anpasst. Die Spezifikation CORBA hat den großen Vorteil, dass sie nicht auf eine Programmiersprache oder ein Betriebssystem festgelegt ist, sondern in fast jeder Programmiersprache realisiert werden kann. Mitte bis Ende der 90er Jahre wurde von Microsoft eine Alternative zu CORBA veröffentlicht: das Microsoft Distributed Component Object Model (DCOM). DCOM ist aber erst ab dem Betriebssystem Windows 95 verfügbar und beschränkt sich ausschließlich auf Microsoft-Betriebssysteme. Microsoft hat außerdem im Jahr 2000 mit .NET („Dot-Net“) eine zusätzliche Technologie veröffentlicht und neben Suns Remote Method Invocation (RMI) noch eine Alternative zu CORBA geschaffen. Im Folgenden wird das herstellerunabhängige Architekturmodell mit CORBA und die herstellerabhängige Architektur .NET von Microsoft näher beschrieben.

### 3.3.1 Verteilte Anwendungen mit CORBA

Lange bevor das World Wide Web (WWW) und Java den Bedarf an verteilten Anwendungen erkennen ließen, hatte die OMG mit CORBA eine Spezifikation verabschiedet, die die Interaktion von verteilten Objekten beschreibt. In der OMG sind z.Zt. ca. 800 Firmen vertreten, die den offenen Standard gemeinsam definieren. Die CORBA-Objekte können derzeit in den Programmiersprachen C, C++, Ada, Cobol, Smalltalk und Java realisiert und auf den Plattformen Windows 95/98/XP/NT/2000, OpenVMS, Linux und verschiedenen UNIX-Derivaten zur Verfügung gestellt werden. Somit ist beispielsweise gewährleistet, dass ein Java-Programm, das auf einem Windows 2000-Rechner ausgeführt wird, über ein Netzwerk auf C++-Objekte zugreift, die von einem Linux-Web-Server angeboten werden. Neben der Sprachen- und Systemunabhängigkeit zeichnet sich CORBA auch als Basis für eine moderne Client/Server-Architektur aus. Der Software-Entwickler muss sich nicht mehr um die Netzwerkkommunikation auf Protokollebene kümmern, da dies durch eine der Hauptkomponenten der Spezifikation definiert ist. Diese von der OMG zuerst defi-

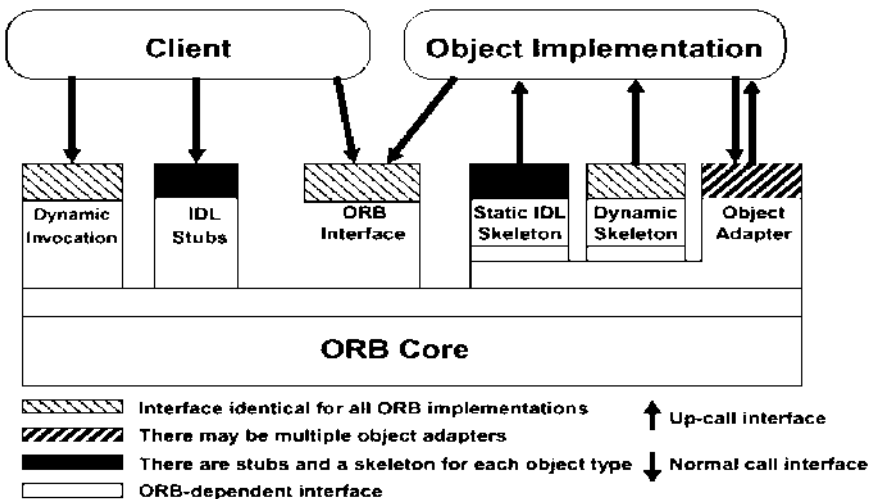
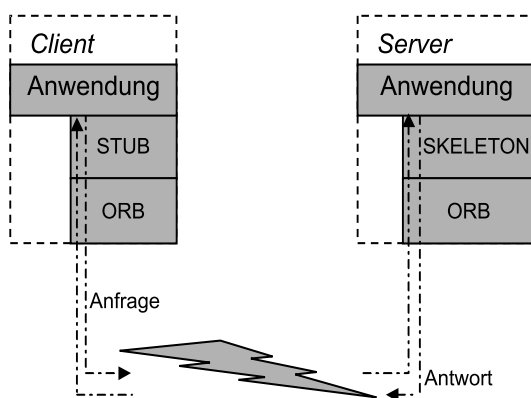


Abbildung 3.4. CORBA-Architektur der OMG

nierte Komponente ist der Object Request Broker (ORB), der unter anderem den Kommunikationskanal zwischen den CORBA-Objekten darstellt. Weitere Komponenten sind die CORBA-Services, die CORBA-Facilities und die Business-Objects. Ab der Version 3.0 wurde zusätzlich das Komponentenmodell definiert, welches am Ende des Kapitels kurz beschrieben wird.

Mit dem ORB wurde der Mechanismus definiert, der die Anfrage eines Clients an ein im Netz befindliches Objekt weiterleitet, unabhängig von dem Ort und dem Betriebssystem, auf dem der Client oder das Objekt realisiert wurden. Die Anfrage besteht auf dem Client aus einem Methodenaufruf mit Parametern, die durch den ORB in einen binären Datenstrom umgewandelt und anhand des GIOP/IIOP an den Server übermittelt werden. Der Server empfängt durch den installierten ORB den Datenstrom, dekodiert diesen und führt den Methodenaufruf aus. Das Ergebnis und die Rückgabewerte des Aufrufes werden auf demselben Weg (codieren/decodieren) an den Client übermittelt.

Die Methoden, die auf dem Server zur Verfügung stehen, werden durch die Stubs auf der Client-Seite und Skeletons auf der Server-Seite in der Interface Definition Language (IDL) beschrieben und in dem Implementation-Repository registriert. Die Begriffe Server und Client sind hier sehr genau zu betrachten, da es möglich ist, dass der Client gegenüber anderen „Teilnehmern“ auch einen Server darstellen kann. Der Begriff Client bezeichnet in diesem Zusammenhang ein Modul in einem Netzwerk, welches eine Anfrage stellt. Der ORB ist der Hauptbestandteil von CORBA, ohne die Installation eines ORBs kann keine verteilte CORBA-Anwendung entwickelt werden.



**Abbildung 3.5.** Kommunikation über ORBs

Die Codierung und Decodierung für die Kommunikation zwischen dem Client und den Objekten wird durch das ORB-Interface sichergestellt, welches die Funktionen für die lokalen Dienste bereitstellt. Die ORB-Architektur besteht auf der Client-Seite aus vier Komponenten (vgl. Abbildung 3.4).

Der Client verwendet die Dienste des Server-Programms durch den Aufruf einer Methode aus der Object-Implementation. Er kann den Methodenaufruf eines entfernten Objektes ausführen, ohne die Netzwerkcommunication kennen zu müssen. Über den IDL-Stub wird eine Schnittstelle zu den Objektdiensten zur Verfügung gestellt, welche in der IDL definiert und von einem IDL-Compiler generiert wurde. Der Stub enthält somit den Quellcode für die Codierung der Methodenaufrufe mit den dazugehörigen Parametern in den Nachrichtenströmen, die an den Server weitergeleitet werden. Um die vom Server zur Verfügung gestellten Dienste lokalisieren und verwenden zu können, wird das Dynamic Invocation Interface (DII) bereitgestellt. Hierfür wurden Standard-APIs in CORBA definiert, die das Methoden-Handling ermöglichen. Eine weitere API steht als Interface Repository zur Verfügung. Diese stellt eine verteilte Laufzeit-Datenbank dar und beinhaltet die codierten Versionen der in IDL definierten Schnittstellen. Eine API ermöglicht das Lesen und das Bearbeiten dieser Metadaten.

Den Client-Komponenten stehen die fünf Komponenten auf dem Server gegenüber (vgl. Abbildung 3.4). Die Object Implementation wird durch die für den ORB angebotenen Programmiersprachen realisiert und durch die IDL-Interfaces beschrieben. Durch die Vielzahl möglicher Programmiersprachen kann die für die zu realisierende Methode am besten geeignete Sprache gewählt werden.

Die IDL-Skeletons sind statische Schnittstellen zu den IDL-Stubs auf dem Client. Sie beschreiben den Dienst, den der Server zur Verfügung stellt, und stellen den Code bereit, der den eingehenden Datenstrom in einen Methodenaufruf wandelt (Decodierung). Sind für die Methode Rückgabewerte definiert, dann werden diese durch den IDL-Skeleton in einen binären Datenstrom gewandelt (Codierung) und an den aufrufenden Stub (Client) zurückgeschickt. Neben dem statischen Skeleton ist auch ein dynamisches Skeleton Interface implementiert. Das Dynamic Skeleton Interface (DSI) bearbeitet eingehende Methodenaufrufe, für die kein statisch IDL-kompiliertes Skeleton vorhanden ist. Das Interface analysiert eingehende Methodenaufrufe und sucht das zu dem Aufruf passende Objekt. Für die auf dem Server implementierten Objekte wurden die beiden Komponenten Object Adapter und Implementation Repository realisiert. Die Dienstanfragen der Clients werden von dem Object-Adapter entgegengenommen, der Object-Adapter weist die Anfrage einem Server-Objekt zu und stellt zusätzlich eine Umgebung bereit, in der die Server-Objekte ausgeführt werden können. Die Objekte, die vom Adapter unterstützt werden, sind in der Laufzeitdatenbank des Implementation Repository registriert. Zusätzlich enthält das Repository Trace-Informationen, Prüfprotokolle und administrative Daten, die im Zusammenhang mit der Implementierung des ORBs stehen.

Neben der Vermittlungskomponente enthält der ORB die CORBA-Services, welche die transaktionsorientierte Abarbeitung der Methodenaufrufe, das persistente Abspeichern von Objekten, Sicherheits-Mechanismen und den Zugriff auf verteilte Objekte über einen Namensdienst zur Verfügung stellen. Die CORBA-Services werden zur Bereitstellung systemnaher Funktionalitäten für die Server-Objekte benötigt. Da die Services ein Bestandteil der CORBA-Definition sind und die ORBs für unterschiedliche Systemplattformen angeboten werden, ist die Verwendung der system-

nahen Funktionen für den Architekten und den Entwickler gleich, da der ORB die unterschiedlichen Plattformen kapselt. Die OMG definiert eine ständig wachsende Anzahl von CORBA-Services, welche auf der Homepage der OMG zu finden sind.

Die CORBA-Facilities werden zusammen mit den Business-Objekten verwendet. Die Facilities werden von Entwicklern verwendet, die die Endanwendung realisieren. Sie stellen die für die Entwicklung benötigten Common Facilities zur Verfügung. Darunter fällt das Drucken, das Versenden und Empfangen von E-Mails, der Datenaustausch und die Internationalisierung. Um eine Anwendung effektiv realisieren zu können, wurde neben den dafür definierten Software-Entwicklungsregeln auch ein Framework für die Business-Objekte definiert. Die Business-Objekte beschreiben im objektorientierten Sinne fachliche Bestandteile des Geschäftsprozesses wie beispielsweise Personen, Adressen, Autos oder Versicherungen, welche in der Anwendung benötigt werden. Die Objekte müssen in der Lage sein, miteinander semantisch verknüpft zu werden, um sich gegenseitig Attribute übergeben und so mit anderen Business-Objekten interagieren zu können. Die dafür notwendigen Schnittstellen müssen sorgfältig und weitsichtig spezifiziert werden, damit ein hoher Grad an Wiederverwendbarkeit erreicht wird und die Business-Objekte auch unabhängig voneinander realisiert werden können.

Ein Business-Objekt stellt seine Dienste für die Clients über IDL-Beschreibungen zur Verfügung. Diese dienen als einheitliche Schnittstellenbeschreibung für die verteilten CORBA-Objekte. Die IDL ähnelt in ihrer Syntax der Programmiersprache C/C++. Alle Objekte, die für den Client sichtbar sein sollen, werden durch das IDL-Interface beschrieben. Die Schnittstelle enthält den Methodenaufruf, die zu übergebenden Parameter und die eventuellen Rückgabewerte.

Nachdem nun alle Hauptbestandteile inklusive deren Komponenten der OMG-CORBA-Architektur beschrieben wurden, folgt die Darstellung der beiden verwendeten Protokolle, die die Kommunikation zwischen den ORBs sicherstellen sollen. Eine verteilte CORBA-Anwendung besteht aus mindestens zwei ORBs, die über ein Netzwerk miteinander verbunden sind. In diesem Fall ist ein ORB auf einem Server und ein weiterer auf einem Client installiert. Durch die ORBs und die enthaltenen Komponenten können die Anfragen empfangen und die angesprochenen Dienste ausgeführt werden. Die Ergebnisse der Dienste werden dann über den ORB auf dem Server an den ORB des Clients zurückgeschickt. Die Kommunikation zwischen den ORBs wird durch die Definition der beiden Protokolle General Inter-ORB Protocol (GIOP) und Internet Inter-ORB Protocol (IIOP) sichergestellt. Das GIOP spezifiziert die Nachrichtenformate, und die Datendarstellung für die Kommunikation und das IIOP definiert den Austausch der GIOP-Nachrichten über ein TCP/IP-Netzwerk. Dank der Standardisierung der Protokolle können die ORBs von verschiedenen Herstellern miteinander kommunizieren.

### **Anmerkung des Autors**

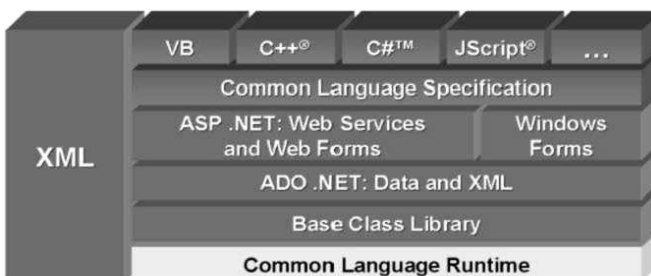
Der hier beschriebene CORBA-Standard basiert auf der Version 2.0. Zurzeit ist die aktuelle Version 3.0. Sie lehnt sich u.a. mit dem implementierten Containermodell stark an J2EE an. Da hier aber nur ein grundlegender Überblick über

CORBA gegeben werden soll, wurden Teile von CORBA nur oberflächlich beschrieben. Für eine detaillierte Betrachtung bedarf es weiterer Literatur oder eines Besuchs auf den offiziellen OMG-Internet-Seiten ([www.omg.org](http://www.omg.org)).

### 3.3.2 Verteilte Anwendungen mit Microsoft .NET

Microsoft verfolgt mit der .NET-Technologie drei Ziele: Zum einen eine Programmiersprachenunabhängigkeit durch das Interpretieren des Sourcecodes. Für diesen Zweck wird der Sourcecode für die Microsoft Common Language Runtime (CLR) in einen Bytecode übersetzt. Diese Runtime-Umgebung ermöglicht es, dass der Sourcecode in den Sprachen Visual Basic, C#, C++ und J# geschrieben werden kann. Ganz gleich mit welcher Programmiersprache der Sourcecode realisiert wurde, das Ergebnis des Compilers ist immer ein Bytecode, der es ermöglicht, die erstellten Module wie z.B. Klassen aus einer mit einem anderen Sourcecode erstellten Anwendung aufrufen zu können. Microsoft stellt allen vier angebotenen Sprachen denselben Klassenbibliotheken-Umfang zur Verfügung. Es ist somit möglich, dass für die Realisierung der Module die optimale Programmiersprache verwendet werden kann. Zusätzlich ist es durch das Angebot der unterschiedlichen Programmiersprachen einfacher, die passende Ressource für die Entwicklung der Module zu finden. Ein weiterer Vorteil ist es, dass .NET als Basis für die Realisierung von Web Services dienen kann, die durch ASP.NET effizient entwickelt und betrieben werden können. Mit der unter dem Namen *Microsoft Visual Studio* bekannten Entwicklungsumgebung soll die Entwicklung von .NET-Applikationen in einer der oben genannten Programmiersprachen erleichtert werden. So können Web Services mit einem ähnlichen zeitlichen Aufwand und dem Know-how von z.B. Visual-Basic-Programmierern entwickelt werden.

.NET besteht aus fünf Hauptbestandteilen: dem Framework, den Tools, den Building Blocks, dem Enterprise Server und dem Mobile Device. Diese tauschen im XML-Format Daten zwischen den auf der Basis der Hauptbestandteilen entwickelten Komponenten aus und werden im XML-Format konfiguriert. Zur Unterstützung werden Klassen zur Verfügung gestellt, die die Arbeit mit den XML-Dateien erleichtern.



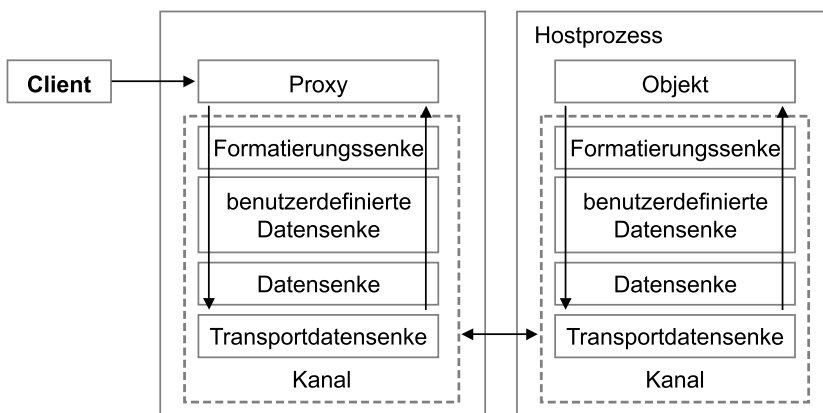
**Abbildung 3.6.** Architektur des .NET-Frameworks<sup>4</sup>

<sup>4</sup> Abbildung aus Saqib Rasool, Microsoft Support WebCast: XML and Microsoft .NET

Es wird nun ein Überblick über die .NET-Architektur gegeben und mit den beiden zuvor beschriebenen Architekturen (J2EE und CORBA) verglichen.

Der allgemein verwendete RPC/RMI-Mechanismus, also der Zugriff auf verteilte Objekte, wird bei .NET Remoting genannt und stellt über das Remoting-Framework die VA-Dienste (verteilte Anwendungsdienste) bereit. Diese VA-Dienste regeln die Activation Lifecycle Services (Lifetime Support), den Zugriff auf Sockets (Channels) und verteilte Aufräumdienste (Garbage Collection).

Die Art der Kommunikation zwischen den Komponenten hängt von den zu verbindenden Komponenten ab. Soll eine Verbindung über einen Kanal (bei .NET Channel genannt), zwischen den .NET-Komponenten aufgebaut werden, so steht ein binäres Protokoll über TCP-Sockets (also ohne SOAP) zur Verfügung. Soll dagegen eine Verbindung zwischen einer .NET-Komponente und einer beliebigen anderen Komponente aufgebaut werden, dann wird diese Verbindung mit SOAP über eine HTTP-Verbindung aufgebaut.



**Abbildung 3.7.** .NET Remoting

Es werden nun die wichtigsten technischen Eigenschaften von .NET zusammengefasst.

**Bedingte Plattformunabhängigkeit:**

Die auf dem .NET-Framework entwickelte Anwendung kann auf den Betriebssystemen verwendet werden, für die Microsoft oder andere (s. das Linux-Projekt Mono) ein .NET-kompatibles Framework entwickelt haben.

**Standards:**

.NET baut auf XML, SOAP und den Webservice-Standards auf.

**Objektorientierung:**

Das .NET-Framework erzwingt keine objektorientierte Entwicklung.

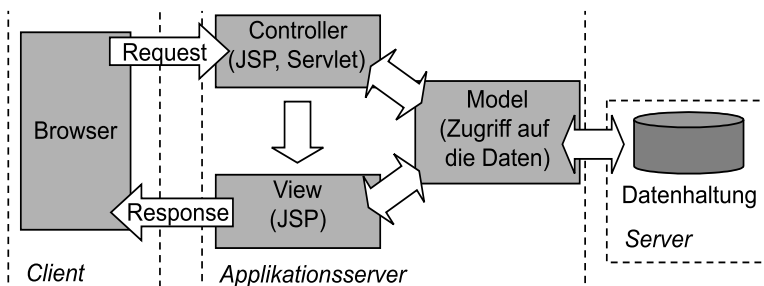
Mehrsprachigkeit:

.NET-Anwendungen können z.Zt. mit den Sprachen C#, J#, C++ und Visual Basic entwickelt werden. Es ist aber durch die Offenlegung von .NET (CLR und CTS) möglich, dass Anbieter weitere Sprachen für .NET entwickeln. Die Mehrsprachigkeit beschränkt sich nicht nur auf die gemeinsamen Klassen des Frameworks, sondern auch auf die einheitlichen Datentypen und damit einheitlichen Aufrufkonventionen.

### 3.4 Kommentar

Die hier genannten Architekturen sind natürlich nicht die einzigen Möglichkeiten, Anwendungen zu realisieren, aber nach Meinung des Autors sind es die am besten geeigneten, um eine investitionssichere und stabile Anwendung zu erstellen. Die Favoriten sind J2EE und CORBA, da beide Technologien sich nicht nur sehr gut ergänzen, sondern auch plattformunabhängig und herstellerunabhängig sind. In diesem Zusammenhang wird als eine besondere Architekturform in Verbindung mit J2EE noch der Model View Controller (MVC) beschrieben. Ein MVC ist ein interaktives System, in dem eine Trennung von Daten, Präsentation und Steuerung der Eingaben erfolgt.

Das System besteht aus den Komponenten Model, View und Controller. Bezogen auf J2EE-konforme Web-Anwendungen beinhaltet das Model die Verbindung zur Datenhaltung und repräsentiert in der Regel die Daten einer Datenbank. Der Controller enthält die Geschäftslogik und nimmt die Anfragen – im Falle einer Web-Anwendung die des Browsers – entgegen. Diese werden dann weiterverarbeitet. Je nach Realisierung kann eine Anfrage das Auslesen der Daten oder das Anzeigen von weiteren Seiten, die im View definiert sind, zur Folge haben. Durch den View werden die Seiten inklusive der Daten an den Browser übermittelt. Der View enthält somit alle Formen der Darstellung von Daten und hat einen Zugriff auf das Model.



**Abbildung 3.8.** Model, View, Controller im J2EE-Kontext

Eine Vielzahl von Architekturen haben bei einer schnell zu realisierenden Anwendung einen höheren Entwicklungsaufwand gegenüber einer .NET-Lösung in einer Microsoft-Umgebung. Dieser zusätzliche Aufwand kann aber aufgrund des höheren .NET-Wartungsaufwands wieder kompensiert werden.



Das große Ziel bei der Entwicklung einer Anwendung ist es, einen hohen Grad an Wiederverwendbarkeit und eine optimale Skalierbarkeit zu erreichen. Das heißt, eine Anwendung ist so zu konzipieren, dass durch einen modularen Aufbau Anwendungsteile von anderen bereits erstellten Anwendungen verwendet werden können. Diese Anwendungsteile sollten dann auch auf dem Server, auf dem sie ursprünglich installiert wurden, ausgeführt werden können.

Leider ist es nicht immer möglich, nur aus rein technischer Sicht eine Anwendung zu entwerfen. Es müssen auch strategische Entscheidungen der Unternehmensleitung berücksichtigt werden. Dies kann zur Folge haben, dass sich z.B. die Unternehmensleitung für eine Microsoft-Partnerschaft entschlossen hat und somit die Entscheidung für eine J2EE-Architektur in Verbindung mit CORBA nicht mehr so einfach möglich sein wird. Auch spielt das vorhandene interne Entwickler-Know-how eine entscheidende Rolle. Müssen erst die internen Entwickler für Java und/oder CORBA geschult werden, schlagen erhebliche Kosten nicht nur für die Schulung zu Buche, sondern auch für das Sammeln der Erfahrungen. In diesem Fall empfiehlt es sich, externes Know-how mit ins Projekt zu holen, um eine „Starthilfe“ für das Projekt zu erhalten. Es muss aber darauf geachtet werden, dass es bei einer Starthilfe bleibt und nicht versäumt wird, das Know-how nach und nach im Unternehmen aufzubauen.

Prinzipiell wird in den nächsten Jahren ein neuer Markt für die Anwender und Software-Architekten entstehen. Dieser ermöglicht, dass die Software-Architekten aus einer großen Auswahl von realisierten Komponenten auswählen und diese in ihre Anwendungen einbauen können. Diese Komponenten wird man direkt aus dem Internet beziehen können: die Komponente liegt auf einem Server im Internet und die Anwender haben eine permanente Verbindung zu diesem Server aufgebaut. In ihrer Anwendung rufen sie eine Methode auf, um die Funktion zu verwenden, und entrichten dafür eine zuvor ausgehandelte Gebühr. Diese Dienstleistung wird unter dem Begriff ASP (Application Service Provider) bereits in einigen Bereichen der Datenverarbeitung angeboten. Die technologische Basis hierfür könnten Web Services sein. Auf diese Technologie wird im Kapitel Ausblick eingegangen. Aber auch J2EE, CORBA und .NET können über ein gemeinsames Protokoll Daten und Nachrichten austauschen.

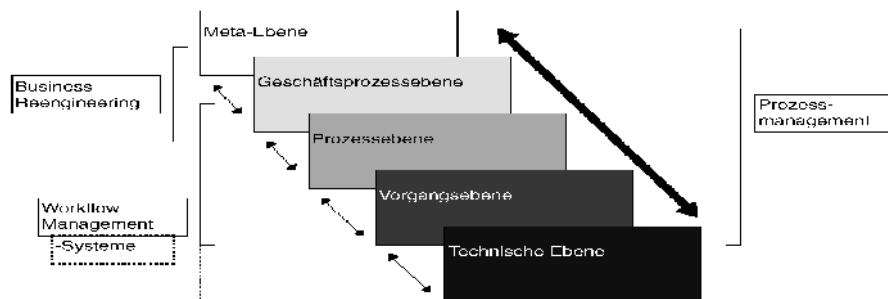
Von diesem neuen Markt werden die Software-Industrie und die Dienstleistungsbranche gleichermaßen profitieren, da durch den Einsatz von bestehenden Komponenten die „time to market“ erheblich sinkt. Die komponentenbasierte Entwicklung von Anwendungen kann in wenigen Jahren dazu führen, dass zukünftig keine größeren Softwareabteilungen mehr in den Unternehmen zu finden sind und dass auf der anderen Seite ein Anbietermarkt entsteht, der sich ausschließlich um die Entwicklung seiner fachlichen Kernkompetenz kümmern muss. Für die Realisierung von Anwendungen werden dann in Zukunft Werkzeuge benötigt, die es ermöglichen, via „drag and drop“ eine Anwendung zu entwickeln, ohne Programmiererfahrung zu benötigen.

## 4 Prozessmodellierung

Dieses Kapitel beschreibt die Modellierung der Geschäftsprozesse. Der Leser wird feststellen, dass die Prozessmodellierung auch in dem Kapitel Workflow-Projekte dargestellt ist. Da dort aber der Zusammenhang mit der Vorgehensweise in Workflow-Projekten im Vordergrund steht, wird in diesem Kapitel die Prozessmodellierung im Detail beschrieben und auf die Verfahren der Modellierung eingegangen.

Die in dem Kapitel Workflow getroffene Unterscheidung der Begriffe Workflow und Prozess wird für eine Positionierung der Prozessmodellierung noch einmal erläutert.

Ein Prozess wird durch formalisierbare Aufgaben wie die Modellierung, die Simulation und die Abwicklung von Prozessen u.a. auch mit Unterstützung von Software für das Prozessmanagement verwaltet. Hinzu kommen die schwer formalisierbaren Aufgaben, die die Unternehmenskultur, die Motivation und die Führung von Mitarbeitern beinhalten. Das Workflow Management dagegen beinhaltet die technikorientierte Planung und Realisierung, den Betrieb und die Kontrolle der Geschäftsprozesse. Für die Durchführung dieser Aufgaben ist eine Unterstützung durch ein WfMS nicht zwingend notwendig. Ein solches System wird aber in der Regel für die Durchführung eines effektiven Workflow Managements eingesetzt. Eine detaillierte Erklärung der beiden Begriffe Prozess- und Workflow-Management findet man in der Literatur (Heilmann 1994). Die Im Folgenden beschriebenen fünf Ebenen orientieren sich an diesem Werk.



**Abbildung 4.1.** Fünf Ebenen der Prozesse (nach Heilmann)

In der *Meta-Ebene* werden die Geschäftsinhalte und Ziele als Grundlage für das Prozessverständnis beschrieben. Auf der Basis der *Meta-Ebene* werden die Kern-

prozesse des Unternehmens grob definiert und in der *Geschäftsprozess-Ebene* zusammengefasst. Die beiden Ebenen enthalten ausreichend Informationen, so dass sie für das Reengineering der Geschäftsprozesse verwendet werden können.

Eine detaillierte Beschreibung der in den oberen Ebenen definierten Prozesse findet in der *Prozess-Ebene* statt. Es werden darin die Abläufe im Einzelnen unter der Berücksichtigung von bestehenden Anwendungen beschrieben. Weiterhin werden Entscheidungen für Anwendungen getroffen, die den Prozess unterstützen sollen.

Die *Vorgangs-Ebene* beinhaltet die Spezifikation der Umsetzung. Durch die Vorgaben aus der *Prozess-Ebene* werden die Abläufe bezogen auf das WfMS und die bestehenden fachlichen Anwendungen spezifiziert, um so die technische Umsetzung der Prozesse im WfMS zu beschreiben. Auf der *technischen Ebene* werden die Lösungen aus den oberen Ebenen durch Programmierung realisiert und implementiert. Die programmtechnische Realisierung erfolgt durch eine technische Anbindung des WfMS an die bestehenden Anwendungen oder durch die Integration des WfMS in die bestehende IT-Landschaft.

In jeder dieser Ebenen muss durch die beteiligten Rollen ein Ergebnis erzeugt werden, welches in den zu den Ebenen gehörenden Dokumenten beschrieben wird. Die Ergebnisse der Ebenen werden in Tabelle 4.1 dargestellt. Man erhält so eine Übersicht, zu welcher Ebene welche Ergebnisse (Dokumente) von welchen Prozessbeteiligten zu erwarten sind.

**Tabelle 4.1.** Ergebnis und Beteiligte der Prozessbetrachtungen

Ebene	Ergebnis	Beteiligte
Meta-Ebene	Ziele des Unternehmens, die mit der Durchführung der Geschäftsprozesse erreicht werden sollen. Diese werden in einem Dokument verfasst und dem Management präsentiert.	oberes Management
Geschäfts-Prozess-Ebene	Prozesslandkarte mit den Kerngeschäftsprozessen des Unternehmens und den an den Prozessen beteiligten Rollen.	oberes Management
Prozess-Ebene	Modellierte Kern- und Hilfsprozesse bzw. Zulieferprozesse, die für die Erreichung der Unternehmensziele notwendig sind.	mittleres und unteres Management
Vorgangs-Ebene	Konzept für die Abbildung der Prozesse in der IT-Umgebung. Diese enthalten die Aktivitäten und Ereignisse der Vorgänge mit den dafür benötigten Ressourcen.	mittleres und unteres Management
technische Ebene	Beschreibt die komplette technische Lösung für die Unterstützung der Prozesse, d.h. die Vorgänge sind in Verbindung mit der IT-Architektur dargestellt.	Sachbearbeiter, Realisierer

Kommt es zu Veränderungen in der Konzeption oder der Dokumentation auf einer der beschriebenen Ebenen, kann dies Auswirkungen auf die benachbarten Ebenen haben. Dies bedeutet, dass bei einer Veränderung innerhalb einer Ebene eine Rückmeldung an die Nachbarebenen notwendig ist. Wird z.B. in der praxisorientierten *Vorgangs-Ebene* festgestellt, dass die zuvor festgelegte Prozessstruktur mit den zur Verfügung stehenden Ressourcen wie beispielsweise den Anwendungen nicht realisierbar ist, muss die darüber liegende Ebene (*Prozess-Ebene*) entsprechend der Machbarkeit verändert werden.

Die Strukturierung der Prozesse in verschiedene Detaillierungsgrade und somit in verschiedene Ebenen vereinfacht nicht nur das Verständnis der Geschäftsprozesse in Unternehmen, sondern auch die Modellierung der Geschäftsprozesse. Während der Modellierung muss darauf geachtet werden, dass die Grenzen und somit der Inhalt der Prozesse nicht verwischt bzw. vermischt wird und die Darstellungen dadurch an Lesbarkeit verlieren. Dieser Gefahr wird durch eine zuvor erstellte Beschreibung der Inhalte für jede Ebene entgegengewirkt.

Durch den unterschiedlichen Detaillierungsgrad werden die Ebenen nicht nur für die Modellierung und Umsetzung, sondern auch für strategische, taktische und operative Entscheidungen verwendet. Die strategische Sicht enthält die *Meta-*, *Geschäftsprozess-* und *Prozess-Ebene* und beschreibt die Prozesse mit dem und für das obere Management eines Unternehmens. Zusätzlich enthält die taktische Sicht auch die *Prozess-Ebene* und die *Vorgangs-Ebene* und beschreibt die Prozesse für das mittlere und untere Management. Die beiden letzten Schichten (*Vorgangs-Ebene* und *technische Ebene*) beschreiben die Prozesse für die Sachbearbeiter des Unternehmens und die Entwickler der Lösung.

Da in der Praxis mit der Prozessmodellierung in der Regel nicht von Anfang an begonnen werden kann, findet man Geschäftsprozesse vor, die aus der Historie gewachsen sind und sich etabliert haben, aber eventuell nicht dokumentiert wurden. Möchte man die Transparenz bezüglich des Informations- und Datenflusses und der vorhandenen Prozesse verbessern, so müssen die Geschäftsprozesse durch unterschiedliche Modellarten beschrieben werden. Die durch Modelle gewonnene Transparenz kann für die Optimierung bzw. die Reorganisation der Geschäftsprozesse herangezogen werden. Ein Modell ist somit ein Abbild der Realität, um diese besser verstehen zu können. In einem solchen Modell werden alle zusammenhängenden Objekte beschrieben. Bei diesen Objekten handelt es sich nicht nur um die Abläufe der Prozesse, sondern auch um die Prozessbeteiligten wie Kunden, Sachbearbeiter usw., und um benötigte Anwendungssysteme.

Die Erstellung eines Modells beginnt mit der Analyse der vorhandenen Geschäftsprozesse. Dies ist entscheidend für den Projekterfolg verantwortlich.

Zur Darstellung der Modelle stehen verschiedene Modelltypen zur Verfügung. Im Folgenden werden die unterschiedlichen Typen kurz aufgeführt.

**Beschreibungsmodelle**

Abbildung des realen Ablaufes in einem formalen System.

**Erklärungsmodelle**

Zusätzlich zum Abbildungscharakter eines Beschreibungsmodells werden im erstellten Modell theoretische Schlussfolgerungen gezogen.

**Entscheidungsmodelle**

Praktische Beschreibung eines Vorgehens zur Findung optimaler Lösungen gut strukturierbarer Probleme.

**Problemlösungsmodelle**

Vorgehensweise zur Lösung schlecht strukturierbarer Probleme, oft unter Verzicht auf das Optimalitätskriterium, statt dessen reicht eine möglichst gute Lösung.

Würde man von Anfang an mit einem hohen Detailgrad der Modellierung beginnen, wäre nach kurzer Zeit die Lesbarkeit eines solchen Modells nicht mehr gegeben. Um dies zu vermeiden, werden die Prozesse, wie oben beschrieben, in unterschiedlichen Ebenen dargestellt und damit in den Modellen durch unterschiedliche Sichten modelliert. Je niedriger die Ebene, desto höher der Detaillierungsgrad. Die Wahl der Sichten hängt – wie bei den Prozessmodellen auch – von dem Ziel der Darstellung ab. Sollen die Unternehmensprozesse modelliert werden, betrachtet man die Kernprozesse oder auch Wertschöpfungsketten des Unternehmens und den Daten- bzw. Informationsfluss zwischen diesen Prozessen. Es wird bei dieser Sicht auf Hilfsprozesse und einzelne Prozessaktivitäten verzichtet. Bei der Beschreibung der Organisationseinheiten werden auch nur die Prozesse genannt, an denen diese Einheiten beteiligt sind, und nicht die einzelnen Aktivitäten. Ziel einer solchen Sicht ist es, einen umfassenden Überblick über die vorhandenen Prozesse im Unternehmen zu erhalten.

Durch die Modellierung wird die Realität in Form eines Modells mit unterschiedlichen Sichten beschrieben. Der Vorteil eines Modells gegenüber einer umgangssprachlichen Beschreibung eines Prozesses liegt darin, dass die Vielseitigkeit der Erklärungsmöglichkeiten gegenüber einer Sprache sehr eingeschränkt ist. Mit einem Modell hat man die Möglichkeit, durch eine zuvor festgelegte Darstellungsform ein System (Geschäftsprozesse) eindeutig zu beschreiben und somit wenig Spielraum für semantische Mehrdeutigkeiten zu geben. Es können auch sehr komplexe Systeme durch die Verwendung von unterschiedlichen Abstraktionsschichten anschaulich beschrieben werden. Diese Abstraktionsschichten ermöglichen die gerade zu Beginn notwendige globale Sicht auf das System, in dem man zunächst die Komponenten zusammenfasst, die erst später im Detail beschrieben werden. Beschreibende Modelle haben entscheidende Vorteile in den Prozess-Entwicklungsphasen *Entwurf*, *Analyse* und *Simulation*. So ist es möglich, mit den am *Entwurf* beteiligten Personen die Kommunikation zu verbessern, indem die Anforderungen des Prozesses anhand eines Modells des realen Systems eindeutig beschrieben werden. Durch die *Analyse* oder auch Prüfung der Modelle können Schwachstellen entdeckt und behoben werden. Sind die Schwachstellen beseitigt,

durchläuft das Modell wieder die Analyse, um dem geforderten realen System sehr nahe zu kommen. Hierbei unterstützt das Modell auch die *Simulation* eines Prozesses. Aufwändig zu erstellende und zum Teil umfangreiche Prototypen können so vermieden werden – das spart Zeit und Geld.

Für eine effektive Modellierung ist darauf zu achten, dass die Modelle den fünf Prinzipien nach Pidd (Pidd 2003) gerecht werden:

1. Model simple – think complicated
2. Be parsimonious, start small and add
3. Divide and conquer, avoid mega-models
4. Use metaphors, analogies and similarities
5. Do not fall in love with data

Wendet man diese Prinzipien als Leitfaden für die Modellierung an, so bleiben durch einfache „Weisheiten“ die Modelle übersichtlich und handhabbar.

Der erste Punkt sagt aus, dass für die Modellierung eine einfache Modellierungsmethode bzw. Darstellungsform verwendet werden soll – dadurch kann man sich mit der eigentlichen Herausforderung besser auseinander setzen. Dabei soll aber trotzdem der komplexe Zusammenhang zwischen den Prozessen dargestellt werden.

Der zweite Punkt skizziert die Vorgehensweise. Es soll mit einer einfachen Übersicht begonnen werden, um auf dieser Basis mit weiteren Modellen den Detaillierungsgrad zu steigern. So können die komplexen Zusammenhänge zwischen den Geschäftsprozessen besser verstanden, aber auch Teilprozesse separat beschrieben werden.

Dies unterstützt auch der dritte Punkt. Hat man einen Überblick gewonnen, so kann mit der Aufteilung und Detaillierung der Geschäftsprozesse begonnen werden und diese können mit bereits erstellten Teilen kombiniert oder von diesen ersetzt werden.

Der vierte Punkt beschreibt die Verwendung von Metaphern, Analogien und Vergleichen zur Beschreibung der Geschäftsprozesse, d.h. sie sollen nicht abstrakt, sondern realitätsnah und verständlich beschrieben werden.

Der letzte Punkt weist darauf hin, dass die zu modellierenden Daten der Geschäftsprozesse nicht im Vordergrund stehen sollen. Es soll erst der Geschäftsprozess in einem Modell mit seinem Kontrollfluss dargestellt werden und danach mit der Modellierung der Daten begonnen werden, die unmittelbar mit dem Ablauf des Prozesses in Verbindung stehen.

Modellierungsmethoden müssen sehr hohen Anforderungen gerecht werden, da sie komplexe Prozesse beschreiben, die für Personen aller Unternehmensbereiche lesbar sein müssen – unabhängig davon, ob diese unmittelbar von dem Prozess be-

troffen sind oder nicht. Zusätzlich ist darauf zu achten, dass die Modellierung implementierungsunabhängig durchgeführt wird. Es dürfen keine Konstrukte von Programmiermethoden oder von Produkten verwendet werden, die diese Unabhängigkeit nicht gewährleisten können. Daraus ergibt sich, dass die Modellierungsmethode allgemein zu halten ist, um die Lesbarkeit der Modelle sowohl durch den Sachbearbeiter als auch den Entwickler zu gewährleisten. Ebenso schwierig gestaltet sich der Umgang mit den unterschiedlichen Sichten. Es empfiehlt sich, zu Beginn der Modellierung, wo man zunächst einen Überblick gewinnen möchte, die identifizierten Teil- und Hilfsprozesse durch entsprechende Platzhalter (Blackbox) darzustellen. Dadurch wird erreicht, dass man sich am Anfang nicht durch Details verzettelt, sondern sich auf das Wesentliche konzentriert. Unterschiedliche Modellierungsmethoden unterstützen dabei unterschiedliche Sichten und Vorgehensweisen. Hier kurz die bekanntesten Modellierungsmethoden im Überblick:

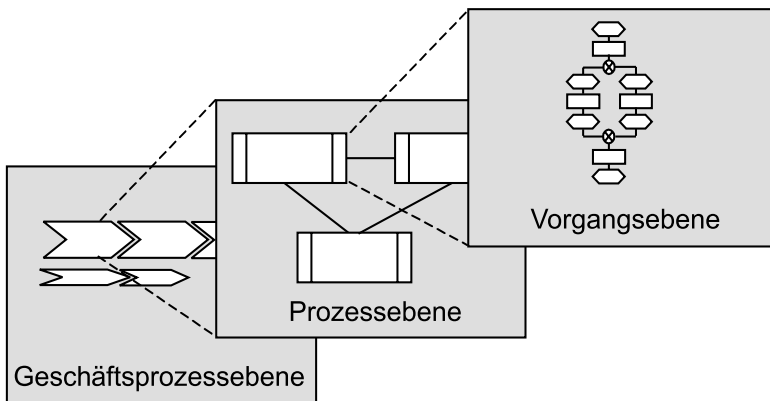
- EPK (ereignisgesteuerte Prozessketten)
- ICN (Information Control Nets)
- OMT (Object Modeling Technique)
- OSSAD (Office Support Systems Analysis and Design)
- PADM (Process Analysis and Design Method)
- Petrinetze
- RAD (Role Activity Diagrams)
- SOM (semantische Objektmodelle)
- UML (Unified Modeling Language)

In dem folgenden Kapitel Modellierungsmethoden werden die Verfahren EPK, eEPK, Petrinetze und UML im Detail erläutert. Es wird beschrieben, welche Abläufe modelliert werden und welche Tätigkeiten durchzuführen sind, um einen Prozess durch ein ausgewähltes Modell darzustellen.

Liegt eine Analyse des Prozesses vor (vgl. Abschnitt 6.2.2 Analyse der bestehenden Prozesse), dann wird mit der Modellierung begonnen. Es empfiehlt sich, im ersten Schritt der Modellierung an einem Modell nur leichte Verbesserungen des Prozessablaufes gegenüber dem realen, ggf. bereits etablierten Geschäftsprozess vorzunehmen. Dieses Modell wird als eine Ursprungsversion des Geschäftsprozesses gesichert, um eine Vergleichsmöglichkeit für weitere Versionen vorliegen zu haben. Wird ein quantitativer Vergleich der zur Zeit implementierten Geschäftsprozesse mit den später optimierten Prozessen benötigt, können Basisdaten dafür durch die Simulation der Ursprungsversion erzeugt werden. Eine Simulation der bestehenden Geschäftsprozesse ist nur erforderlich, falls noch kein WfMS implementiert ist, welches die Laufzeitdaten ermittelt, oder die Laufzeitdaten noch nicht durch eine Analyse der Geschäftsprozesse ermittelt wurden.

Anhand des Ergebnisses der Analyse wird ein grafisches Modell ausgearbeitet, welches einen Überblick über die Kernprozesse und Abhängigkeiten zwischen den Prozessen gibt. Nach der Erstellung eines solchen Modells werden eine Ebene tie-

fer die Kernprozesse und ihre Zulieferprozesse mit einbezogen. Hierfür bieten die Modelle bzw. die entsprechenden grafischen Werkzeuge unterschiedliche Möglichkeiten. Die gängigste Variante ist das Hinterlegen der einzelnen Prozesse hinter den jeweiligen Kernprozessen. Dies wird in einem solchen Werkzeug durch ein weiteres Fenster oder eine zusätzliche grafische Ebene realisiert. In der Praxis empfiehlt es sich, den Inhalt der Fenster an den oben beschriebenen Ebenen (vgl. Abbildung 4.1) zu orientieren. Führt man dies für jede Ebene durch, so erhält man alle Prozesse mit ihren Aktivitäten und den zugeordneten Ressourcen (Rollen wie Anwendungen) für die in der obersten Ebene aufgeführten Kernprozesse. In Abbildung 4.2 und Tabelle 4.1 werden die unterschiedlichen Ebenen und ihre Inhalte aufgeführt.



**Abbildung 4.2.** Ansichten der Ebenen

Ist ein Simulationswerkzeug vorhanden, empfiehlt es sich, dieses zu nutzen und eine Simulation der Prozesse durchzuführen. In diesem Fall muss mit Hilfe der Erfahrung des Prozessverantwortlichen (Processowner) das Prozessmodell mit Gewichtungen versehen werden. Je nach Leistungsumfang des Simulators können aus der Simulation unterschiedliche Kennzahlen gewonnen werden, mit denen eine zielgerichtete Reorganisation der Prozesse möglich wird. Allerdings ist hierbei immer zu beachten, dass eine Simulation nur das Modell simuliert und nie die Realität – d.h. die Simulation ist nur so gut wie das Modell. Die Simulation prüft auch immer nur, ob das Modell konsistent und in sich schlüssig ist – in der Realität können sich dann Abweichungen aufgrund nicht berücksichtigter oder neu aufgetauchter Faktoren ergeben. Die Optimierung von Prozessen durch Reorganisation erfolgt durch eine Veränderung der Abläufe und der personellen Ressourceneinsätze und/oder durch eine effizientere Nutzung der Anwendungsumgebung.

Im Bereich der Ablaufoptimierung können verschiedene Veränderungen durchgeführt werden. Ein stark sequentieller Ablauf kann durch Verzweigungen in mehrere parallele Abläufe aufgespalten werden. Dabei kann es notwendig werden, dass zwischen den parallel laufenden Aktivitäten eine Kommunikation stattfinden muss, um Ereignisse, Zustände oder Daten abzugleichen. Am Ende der Verzwei-



gungen können diese wieder synchronisiert werden. Hierbei ist darauf zu achten, dass es nicht zu Konflikten zwischen den Zweigen kommt. Diese können durch die Modellierung von Ereignissen und Regeln verhindert werden. Wie dies im Detail modelliert und dargestellt wird, hängt von der jeweiligen Methode ab.

Eine besondere Art des parallelen Ablaufes ist die Nebenläufigkeit von Prozessen. Hier haben Aktivitäten, die in unterschiedlichen Zweigen ablaufen, keinen Einfluss aufeinander und können abgeschlossen werden, ohne andere Einzelergebnisse zu beeinflussen und ohne das Gesamtergebnis zu verändern. Bei einer Verzweigung in einen alternativen Verlauf oder einer Nebenläufigkeit kann es zu Konflikten kommen, wenn aufgrund einer Bedingung zwei sich eigentlich ausschließende nachfolgende Aktivitäten ausgeführt werden können. Diese Konflikte werden durch Setzen von zusätzlichen Bedingungen oder Regeln verhindert, und man erhält so wieder einen eindeutigen Verlauf.

Grundsätzlich ist bei der Modellierung darauf zu achten, dass die Prozesse so weit verfeinert werden, bis atomare Aktivitäten vorliegen, die sich fachlich nicht weiter aufteilen lassen. Die Verfeinerung der Aktivitäten ist wichtig, um die Wiederverwendbarkeit sicherzustellen und die Versionssicherheit der Verbindung zwischen Aktivitäten und den dahinter liegenden technischen Funktionen zu erhöhen. Liegen die atomaren Aktivitäten vor, dann kann damit begonnen werden, diesen Aktivitäten die Funktionen zuzuordnen, die in den Fachanwendungen zur Verfügung stehen.

Während der Modellierung werden die Funktionen, die für die Prozesssteuerung benötigt werden, nicht betrachtet: Funktionen beispielsweise für den Start eines Prozesses oder einer Aktivität werden nicht explizit beschrieben, da sie zur Funktionalität des WfMS gehören. Man konzentriert sich auf die Aktivitäten, die für den fachlichen Ablauf der Prozesse erforderlich sind.

Für ein besseres Verständnis der Modellierung werden im folgenden Abschnitt die drei Modellierungsmethoden Petrinetze, EPK und UML beschrieben.

## **4.1 Modellierungsmethoden**

Die Auswahl der drei folgenden Modellierungsmethoden erfolgte anhand deren Verbreitung. Nach Meinung des Autors sind dies die am besten geeigneten Methoden, da beispielsweise die Verknüpfung von EPKs und UML-Diagrammen alle Anforderungen an eine durchgängige Anwendungsmodellierung abdeckt. Der Umfang der folgenden Beschreibung fokussiert auf die Ziele

- dem Leser die Grundlagen der Methoden zu vermitteln sowie
- mit den beschriebenen Methoden einen Geschäftsprozess zu modellieren.

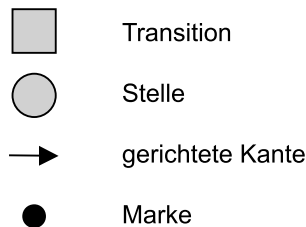
### 4.1.1 Petrinetze

Petrinetze sind Anfang der 60er Jahre aus einer Dissertation von Carl Adam Petri entwickelt worden und dienen zur Darstellung von nebenläufigen, verteilten und asynchron kommunizierenden Prozessen in einem Systemmodell. Die Informationsprozesse werden durch eine mathematisch fundierte Theorie formal beschrieben. Die durch Petrinetze dokumentierten Prozesse weisen folgende Merkmale auf:

- Mit ihnen können die Ereignisse oder Aktivitäten beschrieben werden, die Objekte erzeugen, verwenden oder verbrauchen.
- Sie enthalten die Bedingungen, die für das Eintreten dieser Ereignisse benötigt werden.
- Es können statische und dynamische Systeme modelliert werden.

An diesen Merkmalen kann man erkennen, dass Petrinetze vielseitig anwendbar sind. Sie können für die Modellierung von Daten (relationales Datenmodell), Funktionen und für die Beschreibung des Verhaltens von Systemen verwendet werden. Seit kurzem werden sie auch für die Modellierung von Geschäftsprozessen genutzt.

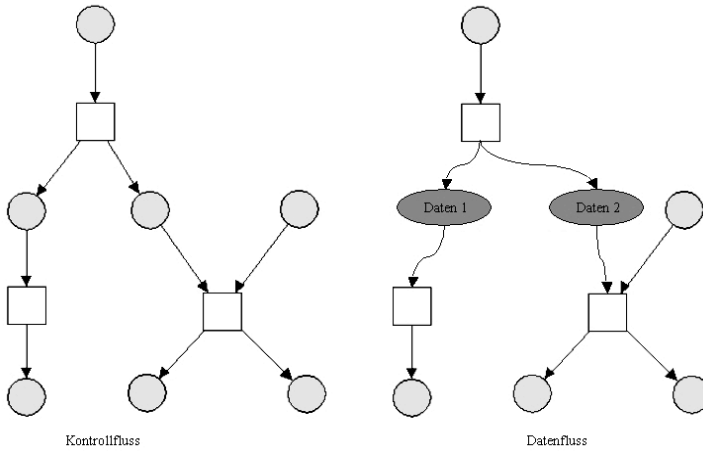
Aufgrund der Möglichkeit, die Realität durch diese Modelle sehr genau beschreiben zu können, besteht die Gefahr, dass Petrinetze schnell nicht mehr lesbar sind. Deshalb ist darauf zu achten, dass Petrinetze in verschiedenen Abstraktionsebenen durch die Nutzung von Filtern vereinfacht werden oder durch Verwendung der höheren Netzklassen (vgl. Tabelle 4.2) Hierarchien gebildet werden.



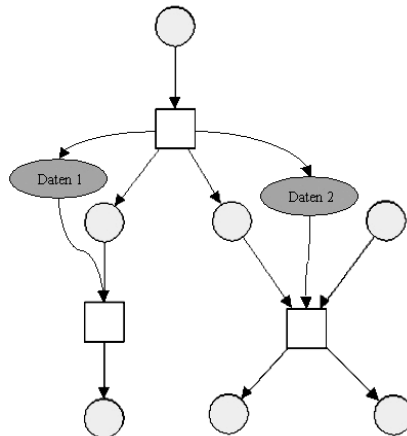
**Abbildung 4.3.** Elemente von Petrinetzen

Für die Darstellung der Modelle werden Transitionen (Aktivitäten), Stellen (Zustände oder auch Speicher), Kanten (Ein- und Ausgaben) und für die Dokumentation von Abläufen Marken (Informationsträger) verwendet. Für eine bessere Darstellung kann der Kontrollfluss und der Datenfluss in zwei getrennten Ansichten modelliert werden. Denn mit dem Kontrollfluss wird der Ablauf des Prozesses anhand der Transitionen und der Stellen dargestellt, und es wird dabei nicht auf die zu bearbeitenden bzw. benötigten Daten eingegangen (vgl. Abbildung 4.4). Bei der Darstellung des Datenflusses entfallen die Stellen, und es werden nur die Transitionen mit den benötigten Daten dargestellt. Durch die gerichteten Kanten wird in einem Datenfluss-Netz definiert, welche Daten in welchen Transitionen benötigt werden (vgl. Abbildung 4.4). Sind beide Informationen gewünscht, kann

man sich an den Transitionen orientieren und die beiden Ansichten übereinander legen (vgl. Abbildung 4.5).



**Abbildung 4.4.** Kontroll- und Datenfluss getrennt



**Abbildung 4.5.** Kombiniertes Kontroll- und Datenfluss

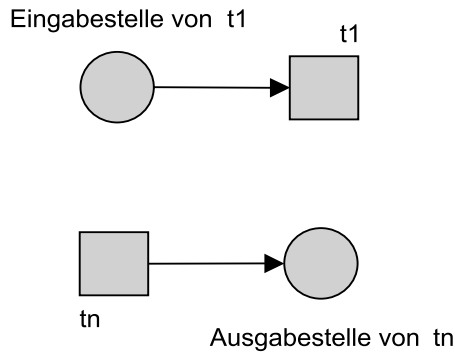
Nachfolgend wird auf die statischen und dynamischen Elemente sowie auf die unterschiedlichen Netzklassen bei den Petrinetzen eingegangen.

### Statische Elemente

Petrinetze bestehen aus Stellen, die durch Kreise visualisiert werden, und aus Transitionen, welche durch Rechtecke oder senkrechte Striche symbolisiert werden. Die Stellen und Transitionen werden durch gerichtete Kanten verbunden. Auf eine Transition folgt immer eine Stelle, und umgekehrt. Es ist möglich, dass eine

Stelle mehrere parallele Transitionen und eine Transition mehrere parallele Stellen als Vorgänger bzw. Nachfolger haben kann. Die Stellen (auch Zustände, Bedingungen, Plätze oder S-Elemente genannt) sind die passiven Knoten in einem Petrinetz, während die Transitionen (auch Ereignisse, Aktionen oder T-Elemente genannt) die aktiven Knoten in den Netzen darstellen.

Die beschriebenen Bestandteile können in bestimmten Kombinationen Eingabe- und Ausgabestellen darstellen (vgl. Abbildung 4.6). Laufen Kanten von einer Stelle zu einer Transition, so handelt es sich um Eingabestellen. Laufen umgekehrt Transitionen zu Stellen, so spricht man von Ausgabestellen.



**Abbildung 4.6.** Ein- und Ausgabestellen

Für die Modellierung mit Petrinetzen werden verschiedene Arbeitsweisen definiert. Durch diese Definitionen können Petrinetze erweitert, verkleinert und optimiert werden. Die folgende Aufzählung entspricht nicht einer Arbeitsreihenfolge, in der die Netze bearbeitet werden müssen, sondern sie ist frei gewählt worden.

Um unterschiedliche Abstraktionsgrade zu erreichen, können Petrinetze feiner oder gröber dargestellt werden. Diese Darstellung unterliegt der Regel, dass zwei Knoten (Stellen oder Transitionen), auch Komponenten genannt, zu einer Stelle oder Transition zusammengefasst werden können, und umgekehrt. Es dürfen nur Knoten zusammengefasst oder verfeinert werden, die an ihrem Rand vom selben Typ sind, also an den Rändern immer aus einer Stelle oder einer Transition bestehen. Durch dieses Verfahren kann ein Modell Schritt für Schritt aus einer sehr abstrakten Sicht immer feiner modelliert werden. Dabei ist darauf zu achten, dass bei der Verfeinerung das geplante Ergebnis des Modells nicht verloren geht. Anders verhält es sich bei der *Einbettung* und der *Restriktion* von Petrinetzen. Hier kann durch das Hinzufügen von Verbindungen und Knoten – z.B. durch eine Einbettung von weiteren Teilprozessen – ein anderes Verhalten des Modells erzielt werden. Bei einer Restriktion dagegen werden Knoten und Verbindungen aus dem Modell entfernt. Eine weitere Bearbeitungsmöglichkeit stellt die Faltung bzw. Entfaltung von Modellen dar. Bei einer Faltung werden gleiche Zweige 1:1 übereinandergelegt. Dabei ist zu beachten, dass alle Strukturen dabei erhalten bleiben.

Das Gegenstück ist die Entfaltung von Zweigen eines Modells; dabei werden die Zweige verdoppelt (vgl. Abbildung 4.9.).

Die beschriebenen Bearbeitungsarten beziehen sich auf die statischen Petrinetze. Eine andere Form der Petrinetze ist das durch ein zusätzliches Element erweiterte dynamische Petrinetz.

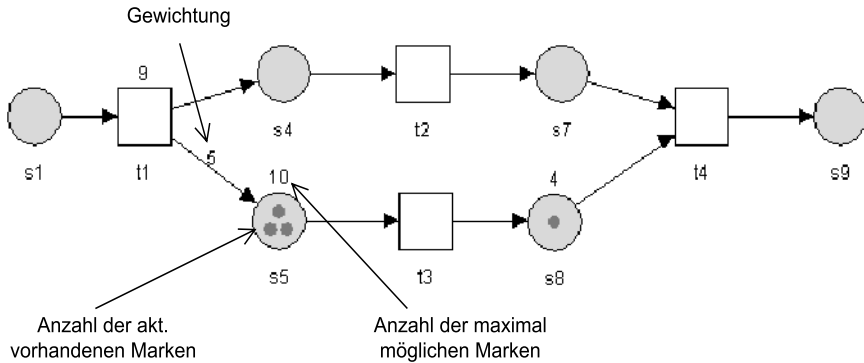
### **Dynamische Elemente**

Mit der Erweiterung der statischen Petrinetze durch das Element Marke ist es möglich, dynamische Systeme (also auch Geschäftsprozesse) zu modellieren und deren Ablauf darzustellen. Die Marke wird grafisch in den Stellen platziert und zeigt durch ihre „Wanderung“ den Ablauf eines Geschäftsprozesses an.

Zur Verfolgung und Simulation eines Geschäftsprozesses gibt es zwei Formen von Marken. Einmal kann keine oder genau eine Marke (boolesche Marken, Ein-Marken-Netz) in einer Stelle vorhanden sein. In der anderen Ausprägung gibt es keine oder beliebig viele Marken in einer Stelle. Der Unterschied liegt darin, dass durch die booleschen Marken lediglich der Ablauf dargestellt werden kann, während mit mehreren Marken in einer Stelle beispielsweise Verarbeitungsprozesse mit ihren Speichern oder Lagern dargestellt werden. Hierbei ist zu berücksichtigen, dass die Verbindungen (Kanten) entsprechend gewichtet werden und die Kapazitäten der Stellen angegeben werden müssen. Die Gewichtung bestimmt die Wahrscheinlichkeit für den Verlauf, den der Geschäftsprozess nehmen wird. In Abbildung 4.7 wird dies beispielhaft dargestellt.

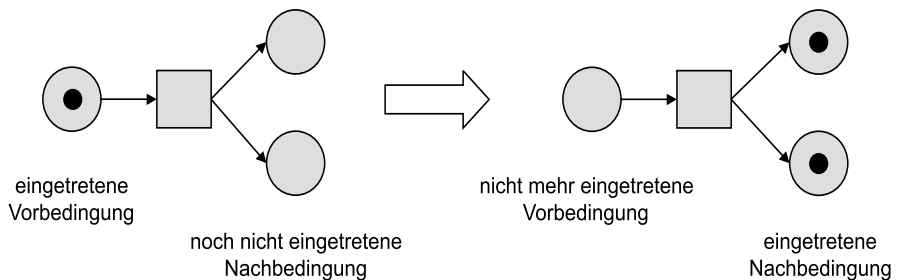
Da das Schalten von Ereignissen in Ein-Marken-Netzen einer Schaltregel unterliegt, kann ein Ereignis nur dann auftreten, wenn es aktiviert wird. Dies ist der Fall, wenn alle Eingangszustände eine Marke enthalten, und alle Ausgangszustände keine Marke enthalten (vgl. Abbildung 4.8). Tritt ein Ereignis ein, so werden die Marken von den Eingangszuständen entfernt und in die Ausgangszustände verschoben. Das Ereignis ist somit eingetreten. Ergänzend können folgende Schaltregeln definiert werden

- Ist eine Vor- und Nachbedingung erfüllt, muss eine Transition nicht sofort schalten.
- Sind mehrere Transitionen zu einem Zeitpunkt schaltbereit, kann nicht davon ausgegangen werden, dass diese gleichzeitig schalten, da in Petrinetzen kein Zeitmaß oder Zeitfluss vorgesehen ist.



**Abbildung 4.7.** Gewichtung von Transitionen und Kapazität von Stellen

Im Lauf ihrer Entwicklung wurden Petrinetze um zusätzliche Elemente erweitert und in Netzklassen unterteilt. Die für die Prozessmodellierung notwendigen Elemente wie AND- und OR-Splits und -Joins (Verzweigungen und Zusammenführungen), Zeitverhalten und weitere Schaltregeln können durch diese Elemente realisiert werden. Unabhängig von der Netzkategorie werden die wichtigsten Elemente im Folgenden beschrieben.



**Abbildung 4.8.** Schaltregeln

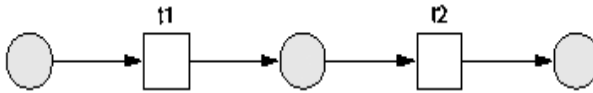
In Tabelle 4.2 werden weitere Eigenschaften und das Verhalten der Netzklassen dargestellt.

**Tabelle 4.2.** Netzklassen

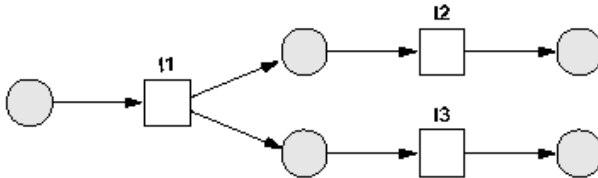
Netzklassen	Beschreibung
S/T-Netze (Stellen/Transition) oder engl. (P/T, Place/ Transition Net)	<p>Die Verbindungen haben eine Gewichtung</p> <p>Verhalten:</p> <p>Die Gewichtungen an den Verbindungen haben Einfluss auf die Aufnahme und Abgabe von Marken der Transitionen.</p> <p>Stellen können 0 bis n Marken enthalten.</p> <p>Die Kapazität K definiert die maximale Anzahl von Marken in einer Stelle.</p>
B/E-Netze (Bedingung/Ereignis)	<p>Transitionen entsprechen den Ereignissen oder Aktionen.</p> <p>Stellen entsprechen den Bedingungen.</p> <p>Marken sind vom Typ Boolean.</p> <p>Verhalten:</p> <p>Jede Stelle enthält eine oder keine Marke.</p> <p>Eine Transition kann nur dann schalten, wenn die Eingabestelle der Transition eine Marke enthält und die Ausgabestelle keine Marke enthält.</p>
Gefärbte Netze (hierarchische oder auch High-Level-Netze)	<p>Durch Farben unterscheidbare Marken für die Modellierung von Attributen und XOR-Verknüpfungen.</p> <p>Zeitbehaftete Petrinetze für verschiedene Zeitkonzepte und zur Performance-Analyse.</p> <p>Mit ihnen können Splits und Joins für AND- und OR-Bedingungen durch spezielle Transitionen dargestellt werden.</p> <p>Es können Hierarchien zur Strukturierung der Modelle verwendet werden. Diese ermöglichen:</p> <ul style="list-style-type: none"> <li>verschiedene Abstraktionsebenen</li> <li>Wiederverwendung von Teilnetzen</li> </ul>
FUNSOFT-Netze	<p>Entwickelt durch die Fraunhofer-Gesellschaft (Institut für Software- und Systemtechnik)</p> <p>Workflow-orientierter Petrinetz-Typ</p> <p>Unterschiedliche Zugriffsstrategien auf Stellen (Kanäle)</p> <p>Es können unterschiedliche Schaltverhalten von Transitionen definiert werden</p> <p>Aktivitäten (Transitionen) können Attribute haben wie z.B.</p> <ul style="list-style-type: none"> <li>Bearbeitungszeit der Aktivität</li> <li>Ausführungsmodus (manuell, automatisch)</li> <li>Anzahl simultaner Ausführungen</li> <li>Verfeinerungsmodus</li> </ul> <p>Formale Semantik durch definierte Abbildung auf P/T-Netze</p>

Für ein besseres Verständnis werden die gängigsten Ablaufstrukturen dargestellt.

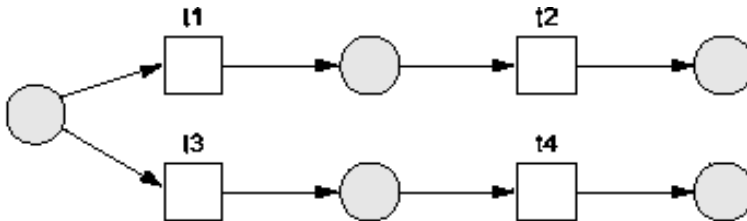
#### Sequenz



#### Nebenläufigkeit

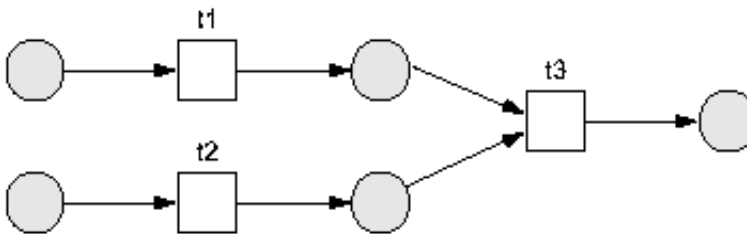


#### Alternative

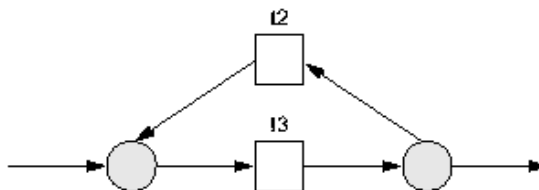


Anmerkung: Durch die Vernachlässigung des Zeitverhaltens kann es zu Konflikten kommen.

#### Synchronisation



#### Iteration





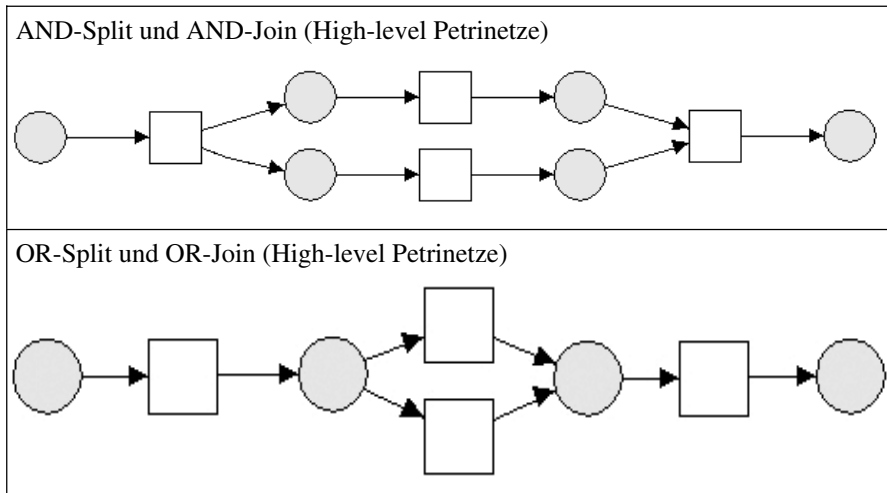


Abbildung 4.9. Ablaufstrukturen

#### 4.1.2 Ereignisgesteuerte Prozessketten (EPK)

Petrinetze sind die theoretische Basis für weitere Modellierungsmethoden, so auch für EPKs, in denen ähnliche Modellierungselemente vorhanden sind. Obwohl EPKs in der Vergangenheit einen hohen Verbreitungsgrad erlangt haben, wurde bisher keine vollständige und konsistente Syntax oder entsprechende Semantik definiert. Eine erste Beschreibung erfolgte mit der Dokumentation des ARIS-Toolsets, in der die Knoten, Sequenzen, Splits usw. beschrieben wurden. Für dieses Buch soll die folgende Definition verwendet werden:

EPK ist eine semiformale und grafische Modellierungssprache, in der die Funktionen und deren gerichtete Verknüpfungen zu Ereignissen einer Prozesskette dargestellt werden. Ereignisse stellen für eine Funktion die Voraussetzung dar, um diese ausführen zu können. Eine Funktion hat mindestens ein Startereignis und erzeugt nach der Bearbeitung wiederum mindestens ein Ereignis, welches das Startereignis wenigstens einer darauf folgenden Funktion ist. Durch eine solche Verkettung wird der Kontrollfluss zwischen den Funktionen modelliert, und es können durch die logischen Verknüpfungsoperatoren alternative oder parallele Abläufe und Schleifen dargestellt werden.

##### 4.1.2.1 EPK-Elemente

Nachfolgend werden die EPKs in Anlehnung an die erweiterten ereignisgesteuerten Prozessketten (eEPKs) beschrieben. Die EPKs sind durch Prof. Scheer um zusätzliche Elemente ergänzt worden und werden als eEPKs im ARIS-Toolset verwendet. Im Anhang findet sich eine Beschreibung des ARIS-Konzeptes bzw. des ARIS-Hauses. Im ARIS-Konzept wird auf die unterschiedlichen Sichten der eEPKs eingegangen, die zur Erhöhung der Lesbarkeit erforderlich sein können.

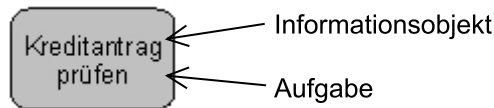
Anfangs werden die Objekte der EPKs beschrieben, um im Anschluss auf die Verknüpfungsarten und die Modellierungsregeln einzugehen.

### Funktionen

Eine *Funktion* ist eine fachliche Aufgabe an einem *Informationsobjekt*. *Funktionen* erhalten Input-Daten, die in ihnen verarbeitet werden, und sie stellen Output-Daten nach außen zur Verfügung. Im Gegensatz zu EPKs erzeugen *Funktionen* im ARIS-Konzept *Leistungen* anstelle von Output-Daten.

Ein Beispiel ist für die *Funktion* „Kreditantrag prüfen“ die erbrachte *Leistung* der „Kreditantrag“. *Funktionen* können so weit unterteilt werden, bis es betriebswirtschaftlich keine Sinn mehr macht.

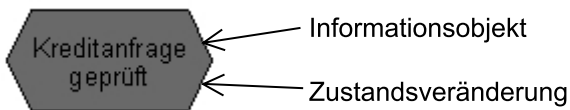
Beispiel: „Kreditantrag prüfen“



### Ereignisse

Mit einem *Ereignis* wird ein (betriebswirtschaftlicher) Zustand des Informationsobjektes beschrieben. *Ereignisse* aktivieren *Funktionen* und werden nach der Ausführung einer *Funktion* erzeugt. Das Ausführen von *Funktionen* wird also durch ein *Ereignis* ermöglicht bzw. setzt das Eintreten eines *Ereignisses* voraus.

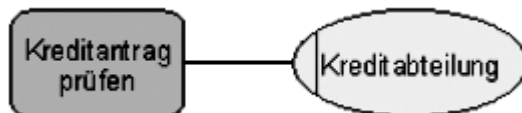
Beispiel: „Kreditanfrage geprüft“



### Organisationseinheiten

*Organisationseinheiten* führen eine *Funktion* aus und sind aufgrund ihrer fachlichen Verantwortung dieser *Funktion* zugeordnet.

Beispiel: „Kreditantrag prüfen“ ==>> „Kreditabteilung“



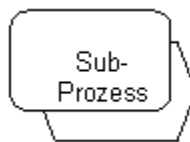
Eine Organisationseinheit kann durch Gruppen und Stellen weiter unterteilt werden.



### Prozesswegweiser

Sie dienen als Verbindung zu weiteren Prozessen und können für den Aufbau einer Prozesshierarchie eingesetzt werden. Im ARIS-Konzept wird der *Prozesswegweiser* als Prozessschnittstelle bezeichnet.

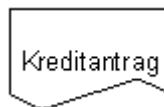
Beispiel: Im Prozess Kreditantrag kann die Schufa-Anfrage ein separater Prozess sein.



### Informationsobjekt

Das *Informationsobjekt* ist ein Element aus der realen Welt und unterstützt die Erreichung von einem oder mehreren Unternehmenszielen. Das *Informationsobjekt* wird für die Beschreibung der *Ereignisse* und der *Funktionen* verwendet.

Beispiel: „Kreditantrag“



Die beschriebenen grafischen Objekte der EPKs werden mit Kanten und Verknüpfungsoperatoren verbunden, um in einem Prozess die verschiedenen Beziehungen wie den Kontrollfluss, den Informationsfluss (Datenfluss) und die Organisationszuordnung darstellen zu können.

#### 4.1.2.2 Beziehungen zwischen den Objekten

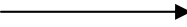
##### Kontrollfluss

Der Kontrollfluss stellt eine gerichtete Verbindung zwischen den Funktionen, Ereignissen, Verknüpfungsoperatoren und Prozesswegweisern dar.

Darstellung: ----->


### Informations- und Materialfluss

Der Informations- und Materialfluss beschreibt durch eine gerichtete Verbindung, ob eine Ressource durch eine Funktion gelesen oder verändert wird.

Darstellung: 

### Organisations- und Ressourcenzuordnung




Durch die Organisationszuordnung werden den Funktionen die Organisationseinheiten zugewiesen, die diese Funktionen ausführen sollen. Zusätzlich können noch Ressourcen zugewiesen werden, die die Bearbeitung der Funktionen unterstützen oder dafür notwendig sind. Dies können Anwendungen oder auch Maschinen sein.

Darstellung: 

### Logische Verknüpfungsoperatoren

Durch die Verknüpfungsoperatoren können logische Verbindungen zwischen Ereignissen und Funktionen dargestellt und die Abläufe in mehrere Teilabläufe unterteilt und dann auch wieder zusammengeführt werden.

**Tabelle 4.3**

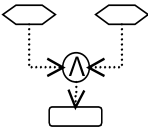
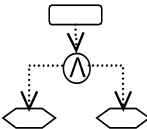
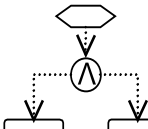
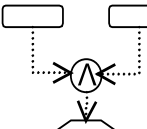
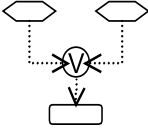
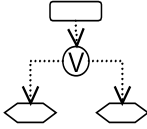
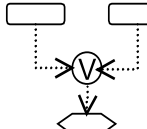
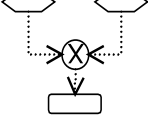
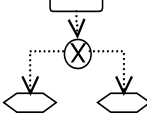
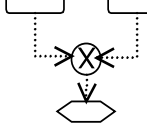
	<i>Trennung des Ablaufes</i>	<i>Zusammenführen des Ablaufes</i>
<b>AND:</b> (Und-Verknüpfung) 	Für den Ablauf werden beide Alternativen durchlaufen (Parallelität)	Es muss von allen Alternativen der gemeinsame nachfolgende Prozessablauf angestoßen werden. Falls eine Alternative den Operator noch nicht erreicht hat, müssen alle anderen auf diese warten (Synchronisation).
<b>OR:</b> (Oder-Verknüpfung) 	Für den Ablauf existieren eine oder mehrere Alternativen	Es wird von mindestens einer Alternative der nachfolgende Prozessablauf angestoßen
<b>XOR:</b> (Entweder-Oder-Verknüpfung) 	Für den Ablauf existiert genau eine von mehreren Alternativen.	Es wird von genau einer Alternative der nachfolgende Prozessablauf angestoßen.

Für die Darstellung einer EPK gibt es keine feste Definition und sie unterliegt einer gewissen Freiheit des Modellierers. Der Modellierer muss sich trotzdem an bestimmte Regeln halten, um in späteren Phasen der Modellierung nicht Probleme bei der Darstellung und Lesbarkeit zu bekommen. Für die Darstellung von EPKs werden deshalb die folgenden Regeln empfohlen.

4.1.2.3 Modellierungsregeln

- EPKs werden von oben nach unten erstellt (vertikal).
- Organisationseinheiten werden rechts neben den Funktionen dargestellt.
- Input-/Output-Objekte werden links neben den Funktionen dargestellt.
- Ein EPK/eEPK beginnt und endet mit einem Ereignis.
- Ein Split kann nur durch den selben Operator wieder zusammengeführt werden.
- Es kann in beliebig viele Pfade verzweigt werden. Es ist aber darauf zu achten, dass mehrere Verknüpfungsoperatoren verwendet werden, um die Logik des Verlaufes darzustellen.
- Auf ein einzelnes Ereignis darf kein OR- oder XOR-Operator folgen, da die Operatoren keine Entscheidung treffen können. In Tabelle 4.4 sind die möglichen Ereignis- und Funktionsverknüpfungen aufgeführt.
- Die Reihenfolge Ereignis – Funktion – Ereignis – Funktion ist einzuhalten.
- Eine Funktion muss mindestens ein auslösendes Ereignis und ein resultierendes Ereignis vorweisen können.
- Jedes Objekt darf nur eine eingehende und/oder ausgehende Beziehung aufweisen.
- Durch eine Kante werden zwei unterschiedliche Objekte miteinander verbunden.
- Eingänge und Ausgänge von Verknüpfungsoperatoren sind immer mit Ereignissen oder Funktionen verbunden.
- Verknüpfungsoperatoren können mehrere Ausgänge oder mehrere Eingänge haben, aber nicht am selben Operator.

Tabelle 4.4. Ereignis- und Funktionsverknüpfung

Verknüpfung	Ereignisverknüpfung		Funktionsverknüpfung	
	auslösende Ereignisse	erzeugte Ereignisse	auslösende Ereignisse	erzeugte Ereignisse
UND				
ODER			nicht erlaubt	
XOR			nicht erlaubt	

Die beiden vorgestellten Modellierungsmethoden, Petrinetze und EPKs, werden der Kategorie der prozessorientierten Modellierung zugeordnet. Eine weitere Kategorie, die objektorientierte Modellierung, wird durch die Unified Modeling Language (UML) repräsentiert. Nachdem Im Folgenden Abschnitt die Grundlagen der UML erklärt wurden, wird anschließend am Beispiel der EPKs und der UML dargestellt, welche gemeinsamen Ansätze die unterschiedlichen Modellierungskategorien besitzen. Führt man beide Modellierungsmethoden zusammen, so kann ein System von der Modellierung der Geschäftsprozesse bis zur technischen Realisierung durchgängig beschrieben werden.

### 4.1.3 Unified Modeling Language (UML)

Die UML ist für die objektorientierte Modellierung von Anwendungen entwickelt worden, um den Projektverlauf von der Entwurfsphase bis zur Implementierungsphase durchgängig zu unterstützen. Der Ursprung liegt in den Vorgängermethoden OOSE, OMT und Booch. Booch, Jacobson (OOSE) und Rumbaugh (OMT) haben die wichtigsten objektorientierten Methoden für die objektorientierten Programmiersprachen wie Smalltalk, C++ und heute auch Java entwickelt. Mitte der 90er Jahre sind alle drei auf unterschiedlichen Wegen Mitarbeiter der Firma Rational geworden und haben 1996 gemeinsam mit der Definition der ersten Version von UML begonnen. 1997 wurde die Version 1.1 bei der Object Management Group (OMG) als Standard eingereicht und veröffentlicht. Weitere Versionen folgten. Heute ist man bei Version 2.0 (Stand Juli 2004) angelangt.

#### 4.1.3.1 Grundlagen

Für ein besseres Verständnis der Beschreibung der UML-Diagrammtypen werden hier noch allgemeine Informationen zu UML gegeben.

Da UML für die direkte Übersetzung des Entwurfs in eine Programmiersprache entwickelt wurde, enthält sie Konzepte für die Systemanforderungs-, Analyse-, Entwurfs- und Implementierungsphasen. Hinzu kommt eine Einteilung in unterschiedliche Sichten, die in Basissicht, statische Sicht und dynamische Sicht unterschieden werden. Mit der Basissicht werden die Objekte (z.B. rotes Auto), die Klassen (z.B. Auto), die Operationen und die Attribute dargestellt. In den statischen Sichten werden dagegen die unveränderlichen Aspekte des Systems beschrieben. Dies erfolgt durch die Darstellung der objektorientierten Struktur (Vererbung, Assoziation, Aggregation und Abstraktion). In den dynamischen Sichten wird das Systemverhalten unter Berücksichtigung der Zeit beschrieben. Im Mittelpunkt steht der Informationsaustausch von Nachrichten, also der Kommunikationsfluss im System. Für eine bessere Beschreibung eines Systems wurden sieben verschiedene Diagrammtypen mit unterschiedlichen Sprachelementen definiert.

**Tabelle 4.5.** UML-Diagrammtypen

Diagrammtyp	Beschreibung
Anwendungsfalldiagramm (Use Case Diagram)	Verbale Beschreibung der Geschäftsprozesse, welche in dem System realisiert werden sollen.
Klassendiagramm (Class Diagram)	Beschreibung des statischen Systemverhaltens über die Klassen mit ihren Beziehungen und Abhängigkeiten.
Paketdiagramm (Package-Diagram)	Bündeln der Klassen in fachliche Module.
Interaktionsdiagramm	Es wird zwischen folgenden Typen unterschieden: a) Sequenzdiagramm (klassenorientierte Aufrufstruktur), welches die zeitliche Abfolge beschreibt b) Kollaborationsdiagramm (nachrichtenorientierte Aufrufstruktur), welches den Nachrichtenfluss (Datenfluss) zwischen den Objekten beschreibt.
Zustandsdiagramm (State Chart Diagram)	Stellt das dynamische Verhalten des Systems dar. Es wird das Verhalten der Objekte beschrieben.
Aktivitätsdiagramm (Activity Diagram)	Beschreibt die Abläufe (Kontrollfluss) des Prozesses
Implementierungsdiagramm	Es wird die (verteilte) Anwendung und die dazu notwendige Hardware beschrieben. Man unterscheidet hier zwischen Komponentendiagramm, welches die Zusammenhänge der Software-Komponenten beschreibt, und Deployment-Diagramm, welches die Hardware-Struktur beschreibt.

Die Diagramme werden in unterschiedlichen Phasen eines Projektes verwendet. Für die Zuordnung der Diagramme zu den Projektphasen wird ein Projekt in die Phasen Analyse, Zielbestimmung, Realisierung/Implementierung, Test und Abnahme eingeteilt. In Tabelle 4.8 wird eine Zuordnung der Rollen zu diesen Phasen nach Seemann und Wolf von Gudenberg aufgeführt.

Werden die UML-Diagramme für die Beschreibung von Geschäftsprozessen verwendet, unterscheidet man zwischen einer internen und einer externen Sicht. Die externe Sicht auf einen Prozess beschreibt den Teil eines Prozesses, der durch jene Prozessbeteiligten bearbeitet wird, die nicht dem Unternehmen angehören. Diese Unterscheidung kann im Anwendungsfall-, Aktivitäten- und Interaktionsdiagramm durchgeführt werden. In der internen Sicht werden durch Paket-, Klassen-, und Aktivitätsdiagramm die Prozessabschnitte beschrieben, die durch die internen Mitarbeiter des Unternehmens bearbeitet werden. Das Aktivitätsdiagramm spielt hier eine Sonderrolle und wird für beide Sichten verwendet. Die Unterscheidung zwischen der internen und der externen Sicht liegt darin, dass der Startpunkt des Geschäftsprozesses bei der internen Sicht einem Akteur zugeordnet wird, der dem Unternehmen angehört. Entsprechendes gilt für die externe Sicht. Die Sicht kann also für denselben Prozess durch die Verlegung des Startpunktes geändert werden.




Die Darstellung und die Elemente der Diagramme für die Beschreibung von Geschäftsprozessen unterscheiden sich in der Darstellung kaum von den herkömmlichen Diagrammen, welche für die Entwicklung von objektorientierten Software-Lösungen verwendet werden.

Da nicht alle Diagramme für die Beschreibung eines Systems zwingend notwendig sind, werden hier auch nicht alle UML-Diagrammtypen bis ins Detail beschrieben. So legen Diagramme wie das Zustandsdiagramm und das Klassendiagramm ihren Fokus auf die Software-Entwicklung.

#### 4.1.3.2 UML-Diagramme

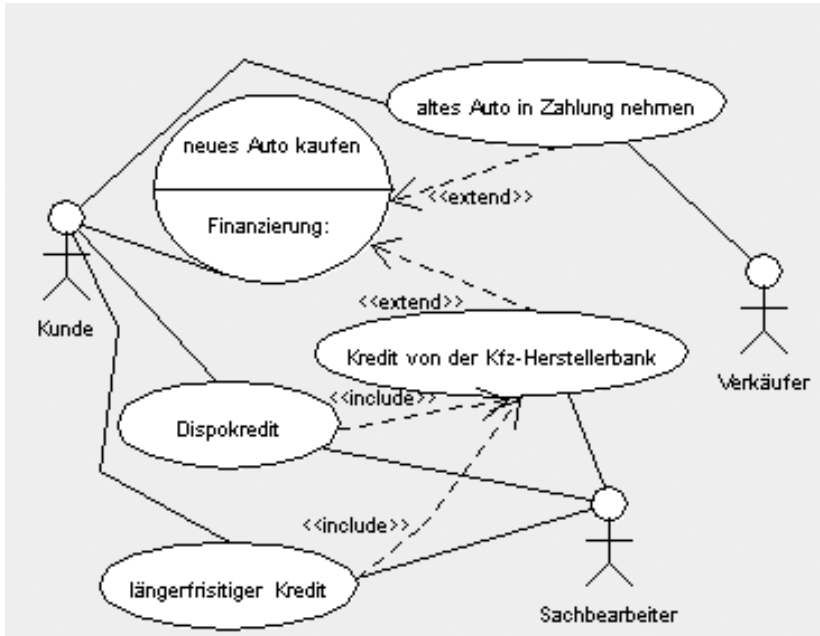
##### Anwendungsfalldiagramm (Use Case Diagram)

Das Anwendungsfalldiagramm beschreibt abstrakt die Interaktionen und Abläufe, also die Eigenschaften einer IT-Lösung bzw. eines Geschäftsprozesses sowie die Schnittstellen zwischen der Außenwelt und dem System. Es wird dabei nicht grundsätzlich davon ausgegangen, dass die Außenwelt eine Person sein muss; es kann sich genauso gut um ein Fremdsystem handeln, welches nicht im Detail beschrieben werden soll. Handelt es sich um Personen, die auf das System einwirken, so muss das Verhalten der unterschiedlichen Rollen dieser Personen beschrieben werden. Das Diagramm gibt einen Überblick über die Zusammenhänge der Kernaktivitäten bzw. Anwendungsfälle in einem Unternehmen und wird durch Szenarios (Use Cases) dargestellt. Hier werden Normalfälle, aber auch Fehlerfälle abgebildet. Diese Use Cases dienen als Basis für die Entwicklung und für den späteren Test der IT-Lösung. Die Darstellung beschränkt sich auf das Element der Akteure, das an dem Geschäftsvorfall beteiligte Personen darstellt, sowie den Anwendungsfall, der durch eine Ellipse dargestellt wird. Die Akteure werden in interne (z.B. Sachbearbeiter des Unternehmens) und externe Akteure (z.B. Kunden) unterschieden und stehen in Beziehung zu den Anwendungsfällen. Die Beziehungen zwischen Anwendungsfall und Akteuren und zwischen den Anwendungsfällen untereinander können von unterschiedlicher Art sein. Man differenziert zwischen einer einfachen Linie und einer gerichteten Linie.

	Stellt die Verbindung zwischen einem Akteur und einem Anwendungsfall dar und beschreibt, dass der Akteur an dem Anwendungsfall beteiligt ist.
	Stellt die Verbindung zwischen zwei Anwendungsfällen dar und beschreibt, dass der Anwendungsfall, auf den der Pfeil zeigt, um den Anwendungsfall erweitert wird, von dem der Pfeil ausgeht.
	Stellt die Verbindung zwischen zwei Anwendungsfällen dar, die ein ähnliches Verhalten und einen ähnlichen Aufbau haben. Die Richtung des Pfeils sagt aus, dass der linke Anwendungsfall ein Teil des rechten Anwendungsfalls ist.



Aus der Beschreibung der Verbindungen geht hervor, dass in einem Anwendungsdiagramm keine Abläufe beschrieben werden. Um das Diagramm besser verstehen zu können, wird in der Abbildung 4.10 ein Beispielanwendungsfall dargestellt, der in den folgenden Diagrammen weiter spezifiziert wird.



**Abbildung 4.10.** Anwendungsdiagramm Autokauf


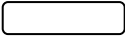

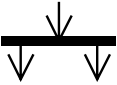
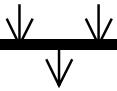


In Abbildung 4.10 wird beispielhaft ein Autokauf in einem Anwendungsdiagramm dargestellt. Neben den Beziehungen zwischen den Anwendungsfällen sind die verschiedenen Rollen (Kunde, Verkäufer und Sachbearbeiter) identifiziert worden, die an den Anwendungsfällen beteiligt sind. Man kann erkennen, dass die beiden Kreditarten („Dispokredit“, „längerfristiger Kredit“) auf Basis „Kredit von der Kfz-Herstellerbank“ aufgebaut sind und dass der Anwendungsfall „neues Auto kaufen“ um „altes Auto in Zahlung nehmen“ erweitert wurde.

Die Use Cases sind aufgrund ihrer Inhalte für Anwender, Analytiker und auch für die Tester von großem Informationswert.

### Aktivitätsdiagramm (Activity Diagram)

Mit dem Aktivitätsdiagramm werden anhand des erstellten Anwendungsdiagramms die Abläufe der identifizierten Anwendungsfälle dargestellt. Dafür werden die einzelnen Anwendungsfälle in ihre Aktivitäten zerlegt und der Fluss der Aktivitäten festgelegt. Durch die Zuteilung der Aktivitäten zu den verschiedenen Akteuren ist es möglich, den Anwendungsfall in der oben beschriebenen internen und externen Sicht darzustellen.

Für die Darstellung des Ablaufes werden die folgenden Elemente benötigt.

	Start einer Aktivität. Es ist nur ein einziger <i>Startpunkt</i> möglich.
	<i>Aktivität</i> in einem Anwendungsfall.
	Durch eine <i>Verzweigung</i> kann aufgrund einer Bedingung der Ablauf einen bestimmten Weg einnehmen. Eine <i>Verzweigung</i> hat einen Eingang und mindestens zwei Ausgänge.
	Mit der <i>Parallelisierung</i> können gleichzeitig bearbeitbare Aktivitäten dargestellt werden.
	Durch die <i>Synchronisierung</i> werden zuvor parallelisierte Aktivitäten wieder zusammengeführt.
	Durch die gerichtete Linie wird die Richtung des <i>Ablaufes</i> vorgegeben. Ein eingehender Pfeil löst eine Aktivität aus und ein ausgehender Pfeil stellt den Abschluss der Aktivität dar.
	Ende einer Aktivität. Es sind mehrere <i>Endpunkte</i> möglich.

Für ein besseres Verständnis wird in Abbildung 4.11 auf Basis des Anwendungsdiagramms „Autokauf“ ein Anwendungsfall beschrieben.

In Abbildung 4.11 sind die Aktivitäten beispielhaft aufgeführt, die die beiden Rollen Käufer und Verkäufer ausführen. Das Diagramm beginnt mit der Auswahl des Pkw und mit der Auswahl des Zubehörs. Hat der Käufer dies abgeschlossen, dann wechseln die Tätigkeiten zum Verkäufer, der die Konfiguration des Pkw bestätigt und die Art der Bezahlung erfragt. Ist der Käufer des Pkw z.Zt. nicht in der Lage den Pkw zu bezahlen, dann wechseln die Aktivitäten in den Prozess (Anwendungsfall) „Kredit von der Kfz-Herstellerbank“. Dieser beginnt mit der Prüfung der Kreditwürdigkeit. Verfügt der Käufer über die nötigen finanziellen Mittel, kann er den Pkw bezahlen.

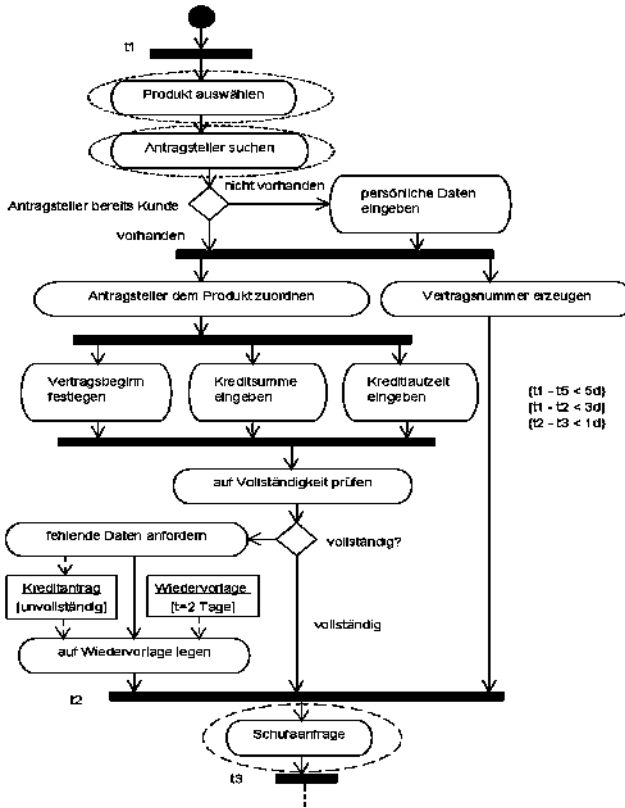


Abbildung 4.11. Aktivitätsdiagramm

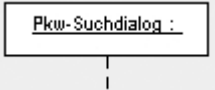



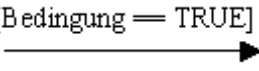
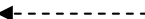
### Interaktionsdiagramm

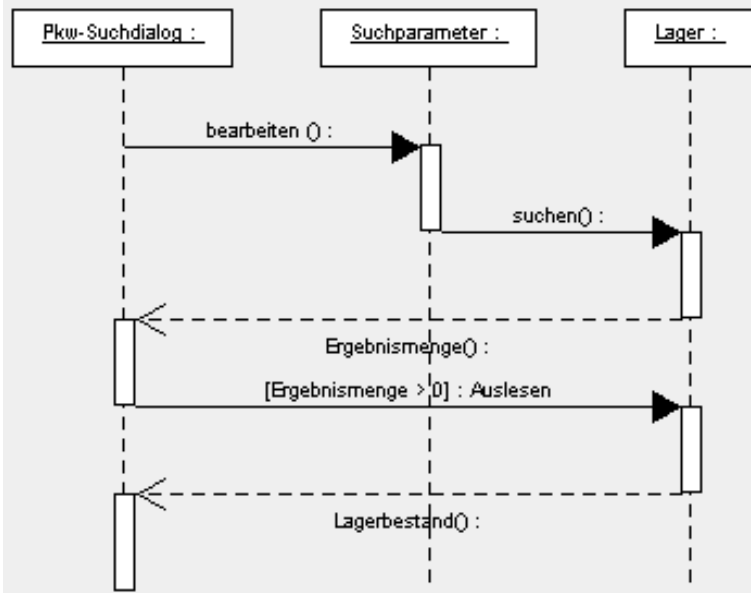
Das Interaktionsdiagramm beschreibt Fallbeispiele (Szenarios) in Verbindung mit den zuvor beschriebenen Diagrammen und den Interaktionen der Objekte. Im Mittelpunkt stehen die Nachrichten, die während des zeitlichen Ablaufs des Anwendungsfalles zwischen den Objekten ausgetauscht werden. Im ersten Schritt der Modellierung wird der Ablauf durch die Annahme eines positiven Verlaufes beschrieben. Im zweiten Schritt folgen die Ausnahmefälle wie beispielsweise Fehlerfälle.

Die Darstellung der Nachrichten kann in Interaktionsdiagrammen durch Sequenz- und Kollaborationsdiagramme erfolgen.

In einem Sequenzdiagramm werden die Nachrichten in Abhängigkeit von den Objekten der identifizierten Klassen dargestellt und beginnen immer mit einem Akteur auf der linken Seite. Der Ablauf des Prozesses folgt der Zeitachse von oben nach unten. Die in einem Sequenzdiagramm vorhandenen Elemente werden anhand der Tabelle 4.6. Sequenzdiagramm Elemente und in einem Beispiel (vgl. Abbildung 4.12) beschrieben.

**Tabelle 4.6.** Sequenzdiagramm Elemente

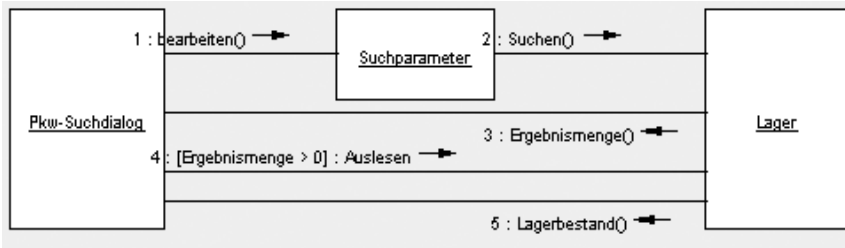
	Objekte, die in der Anwendung Nachrichten senden und empfangen können.
	Senkrechte Linien sind die Lebenslinien der Objekte für den Anwendungsfall.
	Die Rechtecke auf der Lebenslinie zeigen die Lebenszeit der Objekte an.
	Durch die gerichtete Linie wird die Nachricht oder der Aufruf einer Methode angezeigt, welche zwischen den Objekten ausgetauscht bzw. aufgerufen wird.
	Durch die eckigen Klammern werden Bedingungen gekennzeichnet, die in Verbindung mit Nachrichten stehen.
	Durch eine gestrichelte gerichtete Linie werden die Ergebnisantworten und Rückgabewerte dargestellt.

**Abbildung 4.12.** Sequenzdiagramm

In Abbildung 4.12 werden die drei Objekte Pkw-Suchdialog, Suchparameter und Lager dargestellt. Auf einer sehr technischen Ebene wird die Suche nach einem Pkw in einem Lager mit den erforderlichen Methoden und den Ergebnissen beschrieben.

Das Kollaborationsdiagramm beinhaltet dieselben Informationen wie das Sequenzdiagramm und visualisiert die einzelnen Elemente und ihre Beziehungen untereinander. In einem Kollaborationsdiagramm liegt der Fokus auf dem Ablauf und den ereignisbezogenen Aktivitäten zwischen den einzelnen Elementen. Der zeitliche Verlauf der Interaktionen wird durch eine Nummerierung der Nachrichten dargestellt.

In einigen UML-Werkzeugen wird eine automatische Wandlung von Sequenzdiagrammen in Kollaborationsdiagramme und umgekehrt angeboten. Werden wenige Nachrichten und viele Klassen in einem Anwendungsfall benötigt, empfiehlt sich aufgrund der Übersichtlichkeit die Darstellung im Kollaborationsdiagramm. In Abbildung 4.13 ist das Sequenzdiagramm aus Abbildung 4.12 als ein Kollaborationsdiagramm dargestellt.



**Abbildung 4.13.** Kollaborationsdiagramm

Die beiden Formen der Interaktionsdiagramme (Sequenz- und Kollaborationsdiagramm) stellen eine detaillierte Diskussionsgrundlage dar.



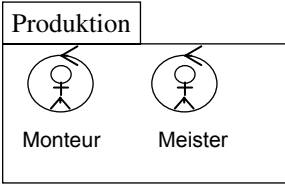
Interaktionsdiagramme werden vor allem von Anwendern, Analytikern, Designern und Sachbearbeitern erstellt und verwendet. Es wird nicht verlangt, dass der Sachbearbeiter Erfahrung im objektorientierten Software-Design vorweisen kann. Seine Aufgabe ist es, den Designer zu unterstützen, indem er die fachlichen Anforderungen an den Designer weitergibt.

### Paketdiagramm

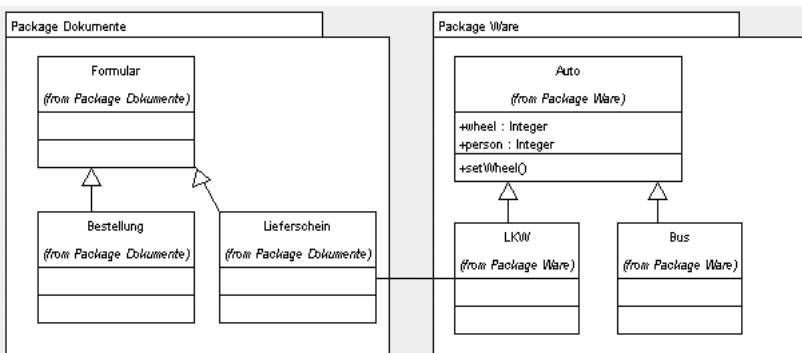
Das Paketdiagramm dient zur Strukturierung der einzelnen Darstellungen von Elementen mit dem Ziel, einen besseren Überblick über das System zu erhalten. Die Pakete werden in Abhängigkeit zu anderen Paketen dargestellt und mit einer gestrichelten gerichteten oder einfachen Linie verbunden. Dies bedeutet, dass bei einer Änderung in einem Paket das Paket an der anderen Seite der Linie eventuell auch eine Änderung erfährt. Pakete ohne eine Abhängigkeit können in einem Paketdiagramm außer Acht gelassen werden.

Im Gegensatz zu den herkömmlichen Paketdiagrammen aus dem Bereich der Softwareentwicklung, in denen Klassen für eine bessere Übersicht in Paketen dargestellt werden, werden in Paketdiagrammen für die Beschreibung von Geschäftsobjekten auch Mitarbeiter und Organisationseinheiten abgebildet. Die Darstellung von Organisationseinheiten ist möglich, da Pakete weitere Pakete enthalten können. Für die Abbildung der Pakete werden neben den oben genannten Linien zusätzlich die in Tabelle 4.7 aufgeführten Elemente verwendet. Ein Beispiel für ein „Klassen“-Paketdiagramm ist in Abbildung 4.14 und für eine Organisation eines Unternehmens in der Tabelle 4.7 zu sehen.

**Tabelle 4.7.** Elemente des UML-Paketdiagramm

Rolle:		Mitarbeiter ist z.B. ein Sachbearbeiter, der einen Kredit bearbeitet.
Geschäftsobjekt:		Geschäftsobjekte, z.B. Kredit, werden durch den Mitarbeiter bearbeitet und durchwandern die modellierten Aktivitäten.
Organisationseinheit:		Eine Organisationseinheit kann Mitarbeiter und Geschäftsobjekte enthalten.

In Abbildung 4.14 sind in den Paketen „Dokumente“ und „Ware“ die Klassen und deren Beziehungen untereinander aufgeführt. Im Paket „Dokumente“ kann man erkennen, dass die Klasse „Lieferschein“ und „Bestellung“ von der Klasse „Formular“ abgeleitet ist. Außerdem steht die Klasse „Lieferschein“ in Verbindung zu der Klasse „LKW“, d.h. für einen Lkw wird ein Lieferschein benötigt.



**Abbildung 4.14.** Klassendiagramm mit Paketen

### Klassendiagramm

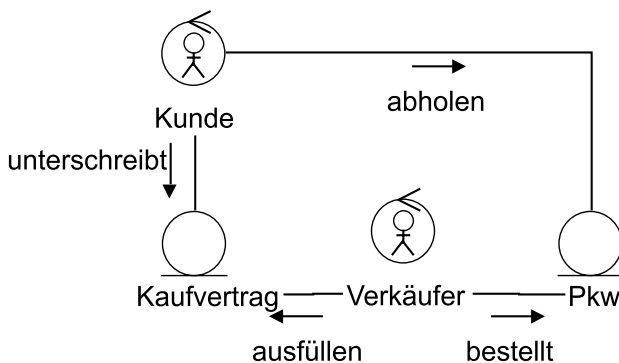
Das Klassendiagramm ist ein zentraler Bestandteil der UML, mit dem man eine statische Sicht auf das System erhält. An dieser Stelle wird zuerst auf die abgewandelte Darstellung für die Prozessmodellierung eingegangen. Diese enthält aber keine grundlegenden Unterschiede in der Darstellung. Es ist in der abgewandelten Form immer noch möglich, dass ein Software-Entwickler mit dieser Struktur ein Klassendiagramm für die Software-Entwicklung erstellt.

Das Klassendiagramm mit dem Fokus Prozessmodellierung stellt die Beziehungen zwischen den Elementen dar, die in dem Paketdiagramm enthalten sind – also die Beziehungen zwischen den Mitarbeitern und den Geschäftsobjekten, die in den Organisationseinheiten zusammengefasst sind. Die Elemente sind bis auf die Beziehungsdarstellung dieselben. Eine Beziehung zwischen den Elementen wird durch eine Pfeilspitze neben einer Volllinie dargestellt. Der Pfeil steht für die Assoziation und zeigt auf das Element, mit dem ein weiteres Element in Beziehung steht.

erstellt ►

Die Beziehung wird in Pfeilrichtung gelesen

In Abbildung 4.15 wird ein Beispiel für ein Klassendiagramm in Anlehnung an das Pkw-Kauf-Beispiel gezeigt. Man kann erkennen, wie die Rollen auf die Objekte Einfluss nehmen. Beispielsweise unterschreibt der Kunde einen Kaufvertrag und holt den Pkw selber ab, den der Verkäufer aufgrund des Kaufvertrages bestellt hat.



**Abbildung 4.15.** Klassendiagramm

Das Klassendiagramm für die Realisierung von IT-Systemen dokumentiert die verwendeten Objekte, Klassen, ihre Methoden, Attribute und die unterschiedlichen statischen Beziehungen der Klassen und Objekte untereinander. Die UML beschreibt aber nicht, wie die Klassen und Objekte ermittelt werden, sondern beschränkt sich auf die Notation und die Semantik für die Darstellung. In einem Klassendiagramm werden Objekte als Konzepte, Abstraktionen oder Gegenstände

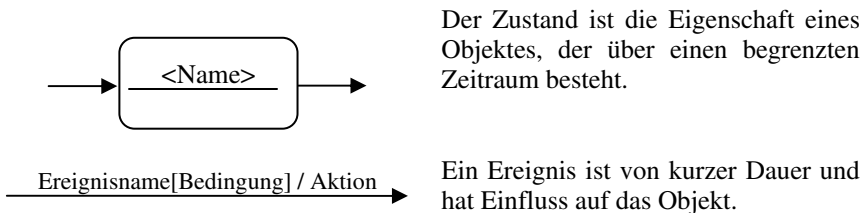
mit einer klaren Abgrenzung und präzisen Bedeutung beschrieben (z.B. blaues Auto). Dagegen werden die Klassen als Gruppen von Objekten mit ähnlichen Eigenschaften beschrieben (z.B. Auto). Objekte und Klassen werden gemeinsam als Rechteck in einem Klassendiagramm dargestellt und unterscheiden sich nur dadurch, dass die Bezeichnung eines Objektes unterstrichen ist. Die identifizierten Klassen werden durch Linien miteinander verbunden, welche die Assoziation, Aggregation, Komposition und Vererbung darstellen.

Eine Klasse besteht aus Attributen und Methoden. Die Attribute beschreiben die Eigenschaften einer Klasse, die anhand der Methoden gesetzt, verändert und ausgelesen werden. Für eine bessere Handhabung der Klassen und ihrer Elemente ist es möglich, die Sichtbarkeit der Klassen, Attribute und Methoden einzuschränken. Man unterscheidet die Sichtbarkeitsbereiche in `public`, `protected` und `private`. Realisiert man die Attribute oder Methoden mit `private`, so haben nur die Methoden dieser Klasse Zugriff auf diese Elemente. Bei `protected` dagegen haben zusätzlich die Objekte einer abgeleiteten Klasse den Zugriff auf die Elemente. Bei `public` sind die Attribute und Methoden für alle sichtbar.

### Zustandsdiagramm

Das Zustandsdiagramm beschreibt das dynamische Verhalten des Systems. Es werden die einzelnen Objekte und ihre Zustände zu einem bestimmten Zeitpunkt im System dargestellt. Der Zustand eines Objektes verändert sich aufgrund von Ereignissen in dem System. Diese Ereignisse haben Auswirkungen auf ein bestimmtes Objekt oder auf eine Gruppe von Objekten. Haben Ereignisse Auswirkungen auf ein bestimmtes Objekt, nennt man diese objektspezifische Ereignisse: sie werden durch ein Objekt ausgelöst und durch ein „Ziel“-Objekt empfangen. Ein objektspezifisches Ereignis kann eine Aktion wie beispielsweise ein Mausklick in ein Datenfeld sein. Der Aktionsname kann ergänzend zu dem Ereignisnamen an die Transition geschrieben werden.

Nachrichten, die von außen durch ein Objekt empfangen werden, nennt man globale Ereignisse. Diese können durch eine Veränderung einer zeitlichen Bedingung oder durch das Ablauf eines Timers eintreten. Für die Darstellung der Zustände und der Ereignisse werden unterschiedliche grafische Elemente verwendet.



Feste Bestandteile in einem Zustandsdiagramm sind der Anfangs- und der Endzustand eines Anwendungsfalles. Die Transition (Übergang) vom Anfangssymbol zum Anfangszustand erhält den Namen des Ereignisses, welches dem Anfangszu-

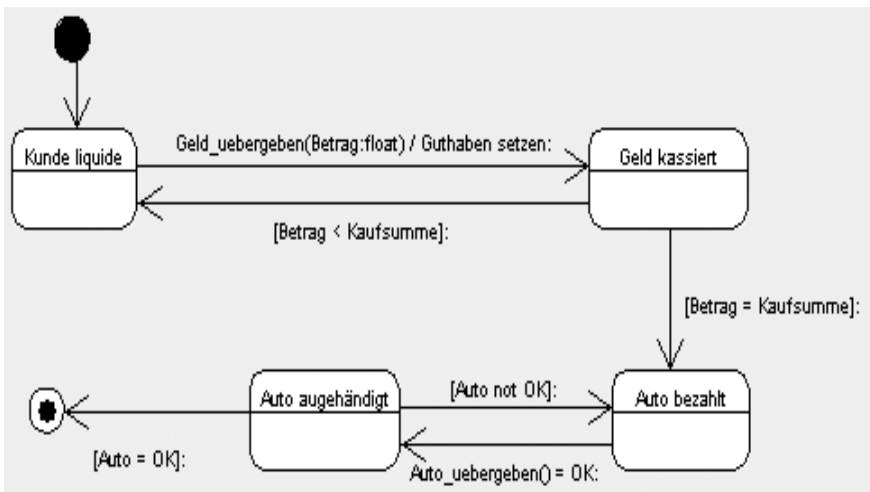


stand zu Grunde liegt. Dagegen erhält die Transition zum Endsymbol den Namen des Ereignisses, welches für das Beenden des Anwendungsfalles verantwortlich ist.



In Zustandsdiagrammen werden die Bedingungen auf den Übergangspfeilen, die von einem Zustand in einen folgenden Zustand führen, mit den Ereignisnamen geschrieben. Die Bedingung muss eine boolesche Bedingung sein, die als Ergebnis wahr oder falsch zurückliefert.

In der Abbildung 4.16 wird anhand eines Beispiels ein Ereignis mit einer Bedingung dargestellt.



**Abbildung 4.16.** Zustandsdiagramm

Die Besonderheit der Zustandsdiagramme sind nebenläufige und hierarchische Teilzustände. Nebenläufige Zustandsdiagramme ermöglichen die Darstellung der Zustände, die ein Objekt zeitgleich erhalten kann. Dadurch ist es möglich, das Verhalten eines Objektes für unterschiedliche Anwendungsfälle aufzuzeigen. Mit den hierarchischen Zustandsdiagrammen können Zustände geschachtelt werden, um so die Lesbarkeit zu erhöhen.

Für detaillierte Informationen über Zustandsdiagramme soll an dieser Stelle auf das Buch Softwareentwurf mit UML von Seemann und Wolff von Gudenberg (Springer-Verlag, 1999) verwiesen werden.

### Implementierungsdiagramm

Das Implementierungsdiagramm enthält die Hardware-Struktur zusammen mit der Sourcecode-Struktur zur Laufzeit der IT-Lösung. Der Fokus liegt dabei auf der Implementierung. Man unterscheidet zwischen dem Komponentendiagramm und dem Deployment-Diagramm. Letzteres beschreibt die Zusammenhänge der Software-Komponenten und stellt die Struktur in Form von Dateien und den dazugehörigen Repositories dar. Es wird von Designern, Software-Entwicklern und Systemadministratoren verwendet. Außerdem enthält das Implementierungsdiagramm ein separates Deployment-Diagramm, welches die Hardware-Struktur beschreibt. Diese Sicht wird nicht nur vom Hardware-Ingenieur erstellt und benötigt, sondern auch vom Software-Entwickler und Designer, um die zu realisierende IT-Lösung in Bezug auf die Leistungsfähigkeit optimal auf die vorhandene Hardware abzustimmen.

Das Implementierungsdiagramm ist für die Realisierung von verteilten Anwendungen besonders wichtig.

#### 4.1.3.3 Zeitlicher Einsatz von UML-Diagrammen

Anhand der Phasenpyramide von Jochen Seemann und Jürgen Wolff von Gudenberg kann der zeitliche Einsatz der Diagramme in einem Projekt abgelesen werden.

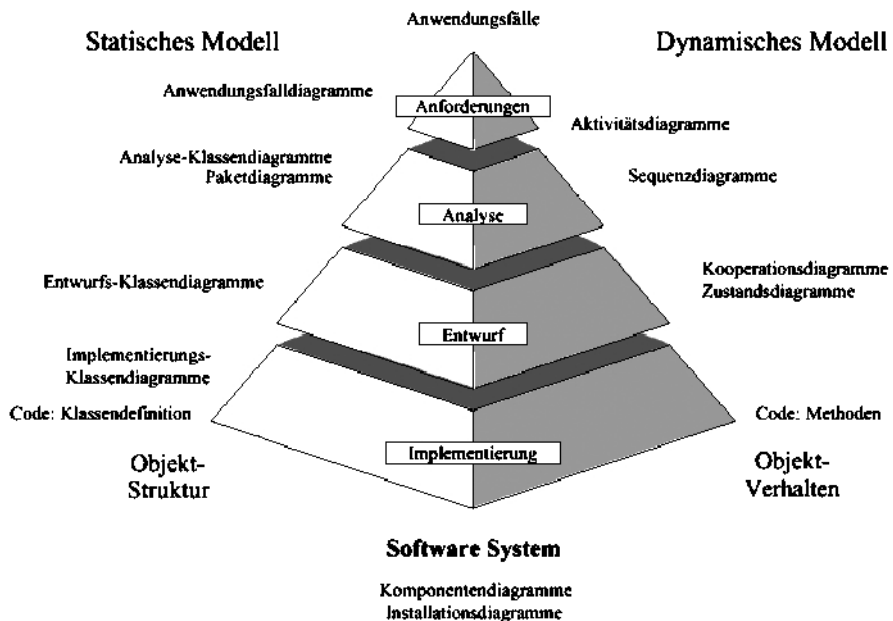


Abbildung 4.17. Phasenpyramide nach Seemann und Wolff von Gudenberg

In der Tabelle 4.8 werden die Rollen zu den Projektphasen und den Diagrammen aufgeführt, welche am Ende einer Projektphase vorliegen müssen. In der Phasenpyramide erfolgt auch eine Einteilung bezüglich des statischen und des dynamischen Verhaltens. So wird klar, mit welchen Diagrammen die Struktur und mit welchen das Verhalten der Objekte dargestellt werden kann.

**Tabelle 4.8.** Zuordnung der Rollen nach Seemann und Wolff von Gudenberg

Phase	Rollen	Diagramme
Anforderungen	Sachbearbeiter, Modellierer, Systemarchitekt	Anwendungsfall- und Aktivitätendiagramme
Analyse	Sachbearbeiter, Modellierer, Systemarchitekt	Klassen-, Paket- und Sequenzdiagramme
Entwurf	Systemarchitekt, Entwickler, Sachbearbeiter (Tester)	Kollaborations- (Kooperations-) und Zustandsdiagramme
Implementierung	Entwickler, Sachbearbeiter (Tester)	Implementierungs- und Paketdiagramme

4.1.4 Gegenüberstellung

Dieser Absatz bietet eine Hilfe bei der Auswahl einer „passenden“ Modellierungsmethode. Die Vor- und Nachteile werden in der Tabelle 4.9 und die grafischen Elemente der Modellierungsmethoden in Tabelle 4.10 zusammengefasst.

**Tabelle 4.9.** Gegenüberstellung der Modellierungsmethoden

Methode	Vorteile	Nachteile
Petrinetze	wenige Elemente	keine einheitliche Notation
	sind visuell gut darstellbar sind vielseitig einsetzbar es wird eine gute Visualisierung des aktuellen Prozesszustandes erreicht es können Einschränkungen simuliert werden können zur Modellierung kooperierender Prozesse verwendet werden	der Detailgrad der unterschiedlichen Ebenen muss durch den Anwender kontrolliert und festgelegt werden
EPK	weit verbreitet	syntaktisch und semantisch noch keine hinreichenden Regeln
	umfassende Darstellungssicht leicht verständlich Standardsoftwarehersteller verwenden EPKs in unterschiedlichen Produkten sind auch in der Software-Entwicklung einsetzbar lassen eine Simulation der Prozesse zu	bieten nur eine statische Sicht auf einen Prozess keine implementierungsnahe Modellierung möglich

Methode	Vorteile	Nachteile
UML	weit verbreitet eignet sich gut zum Informationsaus- tausch zwischen Fachabteilung und Entwicklung umfassende Darstellung Kontrolle zur Einhaltung eines Stan- dards Diagramme können in ihrem Detailgrad wachsen	Keine Zuordnung von Daten zu Akti- vitäten möglich Prozessregeln können nur bedingt ab- gebildet werden eingeschränkte Benutzerfreundlichkeit Verbreitung durch unterschiedliche Dialekte der Produkthersteller

Eine Gegenüberstellung von einzelnen Elementen ist für die unterschiedlichen Modellierungsmethoden nicht einfach, da die Elemente in verschiedenen Diagrammen bzw. Sichten zum Einsatz kommen. In Tabelle 4.10 wird dies durch den Zusatz des Diagrammnamens verdeutlicht.

**Tabelle 4.10.** Grafische Elemente der Modellierungsmethoden

Workflow- Elemente	Petrinetze	EPK	UML
Aktionen	Transitionen	Funktionen	Aktivitätendiagramm
Bedingungen	Stellen	Verknüpfungen in Verbindung mit Er- eignissen	Aktivitätendiagramm, Zustandsdiagramm
Beziehungen	gerichtete Kanten	gerichtete Kanten	dynamische und statische Beziehungen (vgl. Abbildung 4.17)
Zustände	Marken (nur dynami- sche Petrinetze)	Ereignisse	Zustandsdiagramm

## 4.2 Kommentar

Für die Prozessmodellierung ist das Vorgehen wichtig. Am Anfang muss eine grobe Darstellung jener Prozesse stehen, die an der Wertschöpfungskette des Unternehmens beteiligt sind. Im Verlauf der Modellierung kann das Modell bis zu einer detaillierten Beschreibung der Vorgänge verfeinert werden. Die Ergebnisse der Prozessmodellierung dienen nicht nur zur Dokumentation der Prozesse, sondern können auch als Basis für spätere Reorganisationen verwendet werden. Es empfiehlt sich, mit der Modellierung erst dann zu beginnen, wenn die Frage der Werkzeugunterstützung geklärt und die Auswahl des WfMS getroffen ist. Da die Werkzeuge (für Modellierung, Simulation, etc.) sehr eng mit dem WfMS „zusammenarbeiten“, muss eine genaue Abstimmung der Werkzeuge erfolgen bzw. ein Verfahren festgelegt werden, wie die Modelle in das WfMS importiert werden

und wie die protokollierten Laufzeitdaten der Prozesse in das Simulationswerkzeug exportiert werden können. Ebenso sollte sich die Entscheidung für eine Modellierungsmethode nicht nur nach dem Umfang der Möglichkeiten einer Methode richten. Es muss vielmehr auch auf deren Verbreitung bzw. das Know-how der einzelnen an dem Projekt beteiligten Personen geachtet werden. Die drei vorgestellten Modellierungsmethoden sind in ihrer Darstellungsform ähnlich, aber für den Einsatz zur Prozessmodellierung unterschiedlich geeignet. Die Petrinetze stellen die Basis für die EPKs dar und haben ähnliche grafische Elemente. Für die Prozessmodellierung kann zur Darstellung eines Prozesses ein einfaches S/T-Netz verwendet werden, während für die Simulation der Prozesse höhere Petri-Klassen und mächtigere Definitionen verwendet werden müssen. Sonst wird man den Anforderungen zur Darstellung eines Ablaufes nicht gerecht. Hier bieten sich die B/E-Netze oder die gefärbten Petrinetze an, die durch die Marken und die Gewichtung der Verbindungen eine Simulation im Detail zulassen. Eventuell können Mehraufwände entstehen, da ein Wechsel der Petrinetze auch einen Wechsel des Modellierungswerkzeuges zur Folge haben kann. Dies kann man durch die Verwendung der beiden anderen Methoden verhindern. UML-Editoren werden immer mit den UML-Diagrammen angeboten, aber nicht immer mit allen Diagrammtypen. Sie ermöglichen es, die Vorteile der UML zu nutzen. Als Produktbeispiel kann hier *Poseidon for UML* der Firma Gentleware genannt werden. Ähnlich verhält es sich bei den EPKs. Durch die Verwendung des ARIS-Toolsets der Firma IDS Scheer ist es möglich, alle Phasen eines Projektes abzudecken. Falls erforderlich, können die Prozesse auch modelliert werden.

## 5 Workflow-based Integration (Wfbi)

In den vorangegangenen Kapiteln wurden die einzelnen Bestandteile einer prozessbasierten Integration wie Workflow, EAI und Software-Architekturen vorgestellt. Es gilt nun, diese zusammenzufassen, um am Ende eine ganzheitliche Vorgehensweise zu erhalten, die in Kapitel Workflow-Projekte beschrieben wird.

Die prozessbasierte Integration verfolgt den Ansatz, auf Basis von modellierten Geschäftsprozessen die Integration unterschiedlicher Anwendungen zu realisieren. Die technischen Anforderungen der Realisierung werden bei diesem Vorgehen in erster Linie durch die fachlichen Erfordernisse der Geschäftsprozesse bestimmt.

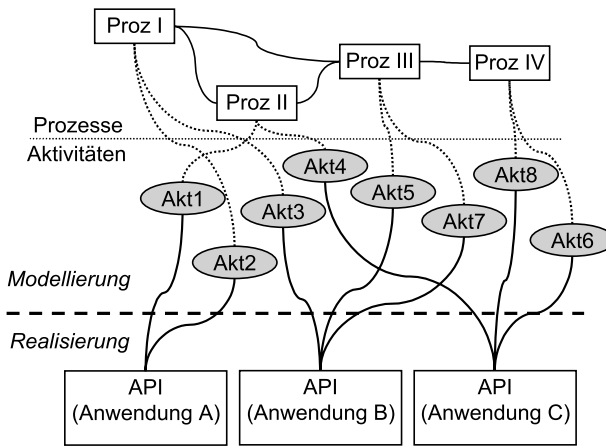
Der Ansatz einer prozessorientierten Ablauforganisation, also die Durchführung von Aufgaben und die Koordination der zeitlichen und räumlichen Abhängigkeiten der Aufgabendurchführung, ist nicht neu. Es wird schon seit mehreren Jahrzehnten angestrebt, diese Abläufe durch vorhandene Technologien umzusetzen und zu unterstützen. Aber erst durch die Entwicklung von leistungsfähigen Technologien in Form von verteilten Rechnersystemen und der Einführung modularer Software-Architekturen ist dies unter ganzheitlichen Lösungsansätzen in großem Rahmen möglich geworden.

### 5.1 Modellierungsansatz

Das Ziel einer prozessbasierten Integration ist die Steigerung der Effizienz auf Basis der Wertschöpfungskette des Unternehmens: durch die Einführung eines Workflow-Managements soll eine Steigerung der Bearbeitungsqualität und eine Beschleunigung der Geschäftsprozesse bei gleichzeitiger Kostenoptimierung erzielt werden. Der grundsätzliche Ansatz für eine Integrationslösung muss es sein, die Prozesse durch ein Modellierungswerkzeug zu beschreiben, um diese in ein elektronisches Format überführen und in ein WfMS überleiten zu können. Dies unterscheidet sich von der traditionellen Modellierung nur darin, dass die Anwendungen, die für die Durchführung des Prozesses als notwendig erkannt worden sind, in Form von grafischen Standard-Objekten eingebunden werden können. Diese Objekte müssen im ersten Schritt der Modellierung keine direkte (technische) Verbindung zu den Anwendungen besitzen, sondern lediglich die Funktionalitäten der Anwendungen grafisch repräsentieren. Es werden so die technischen Insellösungen durch die Modellierung prozessorientiert zusammengeführt, und man erhält einen Überblick über die zu integrierenden Anwendungen.

Die vorausgehende Modellierung der Geschäftsprozesse und die darauf folgende modulare Realisierung zeigt die Trennung bei der Vorgehensweise – diese Trennung von Ablauf und Funktion eines Prozesses muss sich auch in der Architektur widerspiegeln.

Nachdem die an der Wertschöpfungskette orientierten ausgewählten Geschäftsprozesse unabhängig von den Anwendungen und der Betriebsorganisation modelliert wurden, kann damit begonnen werden, die einzelnen Aktivitäten für jeden Geschäftsprozess zu definieren. Sind die Aktivitäten beschrieben, ist es erforderlich, diesen Aktivitäten die benötigten Fachfunktionen der Anwendungen zuzuweisen. Aus diesem Vorgehen ergeben sich die drei Schichten, die in Abbildung 5.1 dargestellt sind.



**Abbildung 5.1.** Beziehungen zwischen Anwendungen und Prozessen

Aus der Abbildung wird ersichtlich, dass die Verbindungen zwischen den Aktivitäten und den Prozessen lockerer sind als die Verbindungen zwischen den Aktivitäten und den Funktionen der Anwendungen. Dies soll verdeutlichen, dass eine Aktivität nicht nur für einen Prozess verwendet werden kann, sondern auch in weiteren Prozessen nutzbar ist. Im Gegensatz dazu steht die Verbindung zwischen Aktivität und Funktion der Anwendung. Eine Aktivität muss immer einen Teil der Funktionalität einer Anwendung repräsentieren, d.h. die Aktivität „Lagerbestand abfragen“ wird immer eine Verbindung mit einem Lagersystem haben, kann aber in den Prozessen „Bestellung“ und „Reklamation“ eines Versandhauses in derselben Form verwendet werden. Nur wenn diese Aufteilung konsequent durchgeführt wird, kann mit einem hohen Grad an Wiederverwendbarkeit der Aktivitäten gerechnet werden. Ein weiterer Aspekt für die Kapselung der Fachfunktionen über Aktivitäten ist der einfachere Austausch einer Anwendung (z.B. Lagersystem). Da der Integrationsserver im Idealfall nur eine fachliche Funktionsgruppe zur Verfügung stellt und diese mit einem darunter liegenden Produkt verbunden ist, kann dieses ohne größeren Aufwand durch ein anderes Produkt ausgetauscht werden, welches dieselbe Funktionalität bereitstellt.

Sind die ersten Prozesse in einem Unternehmen modelliert, so stehen die dafür modellierten Aktivitäten auch noch für weitere zu implementierende Prozesse zur Verfügung. Erstellt man weitere Geschäftsprozesse überwiegend mit Aktivitäten aus diesem Bestand (Aktivitäten-Pool), sind diese mit weitaus geringerem personellen und zeitlichen Aufwand zu realisieren. Gleichzeitig wächst der Aktivitäten-Pool, da einzelne noch nicht vorhandene Aktivitäten im Rahmen der Implementierung neuer Geschäftsprozesse sukzessive realisiert und dem Pool hinzugefügt werden.

Die Trennung der Lösung in die Ablauflogik auf der einen Seite und die Fachfunktionen der Anwendungen auf der anderen Seite hat nicht nur zur Folge, dass die später zu realisierenden Prozesse auf die bereits bestehenden Aktivitäten zugreifen können. Ein weiterer Vorteil ist, dass Veränderungen in der Ablauflogik, also dem Ablauf des Geschäftsprozesses, in der Regel keine Veränderungen auf der funktionalen Ebene zur Folge haben. So kann gewährleistet werden, dass die Geschäftsprozesse innerhalb kürzester Zeit an neue Gegebenheiten des Marktes angepasst werden können. Auf der anderen Seite können fachliche Anwendungen des Produktherstellers A gegen die des Produktherstellers B ausgetauscht werden. Voraussetzung dafür ist, dass das Produkt des Produktherstellers B alle in den Prozessen implementierten fachlichen Funktionalitäten bietet und das Produkt über einen Adapter in die Lösung integriert werden kann. Ist dies der Fall, so kann das Produkt des Produktherstellers A allein durch eine Anpassung des entsprechenden Adapters ausgetauscht werden.

## 5.2 Technischer Ansatz

Der Hauptansatz bei der Realisierung von prozessorientierten Anwendungen ist die Trennung von Ablauflogik (Kontrollfluss) und den Anwendungsteilen bzw. funktionalen Modulen der Anwendungen. Daraus resultiert eine Komponentenbauweise, die durch ihre Architektur einen hohen Freiheitsgrad bezüglich der Verwendung der fachlichen Komponenten aufweist.

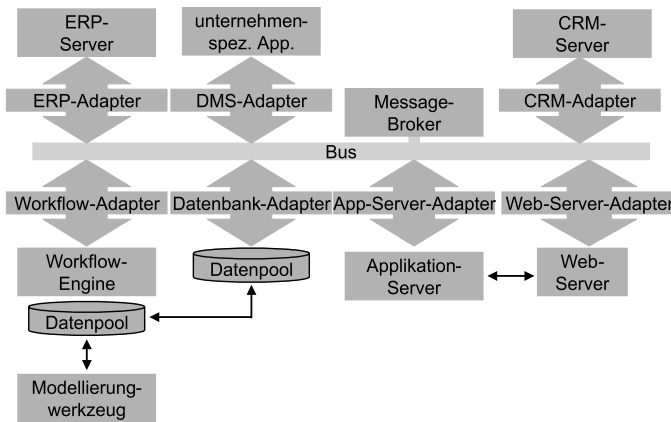
Die WfBI-Architektur verbindet die einzelnen Anwendungen und ihre Komponenten über Adapter mit einem WfMS, einem Abrechnungssystem, mit weiteren Web-Servern, usw. Diese Adapter sind an einem Bus angeschlossen, über den die Nachrichten und Daten ausgetauscht werden. In Abbildung 5.2 erkennt man die beiden Bereiche EAI und Workflow. Außerdem ist zu sehen, dass die Kommunikation zwischen den Anwendungen – neben der Steuerung der Prozesse – der wichtigste Bestandteil ist. Eine solche Architektur kann durch

- eine Produktauswahl jeweils für Workflow und EAI,
- eine Eigenentwicklung auf Basis von CORBA und/oder J2EE,
- die Auswahl eines EAI-Produktherstellers, der ein WfMS mit eingebunden hat,
- und weitere Varianten realisiert werden.



Unabhängig von der Wahl der Variante müssen folgende grundlegende Dinge beachtet werden:

- Die Verbindung (Daten und Nachrichtenübertragung) zwischen den Anwendungen muss mit einem Standardprotokoll oder -format realisiert werden. Nur so kann gewährleistet werden, dass zukünftige Anwendungen und Fremdprodukte effizient angebunden werden können.
- Es ist auf eine performante Anbindung zu achten. Der Wechsel zwischen unterschiedlichen Übertragungsformaten und zwischen unterschiedlichen Architekturen geht zu Lasten der Performance. Können keine theoretischen (mathematischen) Aussagen über das Leistungsverhalten getroffen werden, ist für die Durchführung eines Leistungs- und Lasttests ein Referenzsystem aufzubauen.



**Abbildung 5.2.** WfBI-Bus-Architektur

### 5.2.1 Variante 1

In Abbildung 5.2 ist eine Architektur zu sehen, die unabhängig von Produkten und Technologien eine mögliche Anbindung repräsentiert. Im Mittelpunkt steht der Bus, der in der Realität durch ein Produkt (z.B. einen CORBA-Message Broker) realisiert werden kann. An diesem Bus sind alle Anwendungen über Adapter angeschlossen, die für die Geschäftsprozesse benötigt werden. Auf der Busebene wird ein Modul benötigt, welches den Verkehr der Daten und Nachrichten verwaltet. Dieses Modul empfängt die Daten und Nachrichten, wandelt diese eventuell in ein allgemein verständliches Format um und leitet sie an den jeweiligen Empfänger (z.B. eine Anwendung oder das WfMS) weiter. Die Verwaltung der Nachrichten und Daten kann durch ein entsprechendes EAI-Produkt oder durch eine auf CORBA basierende Eigenentwicklung realisiert werden. Auf einer höheren Ebene wird das WfMS-Modul benötigt, welches die Geschäftsprozesse verwaltet. Dieses Modul kennt die Abläufe der Prozesse und die Funktionen, die hinter den Aktivitäten liegen. Für dieses Modul ist es aber nicht von primärem Interesse, wo sich die Anwendungen befinden, in denen die Funktionen realisiert sind. Dies ist wie-

der die Aufgabe des Message Brokers. Wie man aus der Abbildung erkennen kann, besteht die Möglichkeit, beliebige weitere Module an den Bus anzubinden. Dies kann z.B. ein weiterer Web-Server sein, mit dessen Hilfe Anwendungen über Firmengrenzen hinaus angebunden werden können. Denkbar sind auch zusätzliche Datenbanken zur Realisierung einer gemeinsamen Datenhaltung.

Zur Realisierung dieser Variante liegt es nahe, ein EAI-Produkt mit einem implementierten WfMS zu verwenden. Eine solche Kombination wird z.B. von IBM und BEA angeboten. Aus einem solchen Verbund ergeben sich die folgenden Vor- und Nachteile:

Vorteil:

- einheitliches Kommunikationsformat zwischen WfMS und EAI-Server
- Es kann davon ausgegangen werden, dass die Kombination zwischen WfMS und EAI-Server funktioniert.
- Support durch ein einzelnes Unternehmen und damit ein verantwortlicher Ansprechpartner bei Problemen
- geeignet für Lösungen mit einer hohen Durchsatzrate

Nachteil:

- Abhängigkeit von einem Produkthersteller
- unter Umständen ein nur bedingt offenes System
- Das Kosten-Nutzen-Verhältnis ist erst für eine hohe Durchsatzrate<sup>5</sup> geeignet.

### 5.2.2 Variante 2

Die in Abbildung 5.3 dargestellte Beispielarchitektur zeigt, dass die Daten- und Nachrichtenverwaltung durch das WfMS realisiert wird und deshalb ein separater Bus nicht vorhanden sein muss. Mit dieser Variante kann ebenfalls die Trennung von Ablauflogik und Fachanwendungen erreicht werden, ohne ein EAI-Produkt einsetzen zu müssen.

Vorteil:

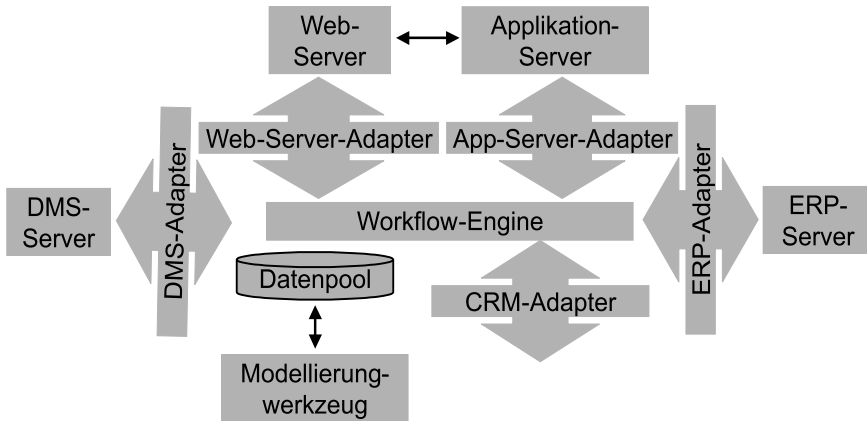
- Freiheit bezüglich der Produktwahl  
Es kann immer das für das Unternehmen optimale Produkt gewählt werden, welches bei einer bereits durchgeführten Kombination von Lösungen (Produkten) nicht gegeben ist.
- geringer Aufwand für die Realisierung
- Geeignet für eine kleine bis große Durchsatzrate durch ein gutes Kosten-Nutzen-Verhältnis.

---

<sup>5</sup> Anzahl der Transaktionen innerhalb der Lösung, welche einen EAI-Server, WfMS und weitere Applikationen enthalten kann.

Nachteil:

- Die Produkte und die Adapter stammen ggf. nicht aus einer Hand. Dadurch können Probleme bezüglich Support und garantierten Leistungen entstehen, da kein verantwortlicher Ansprechpartner bei Problemen zur Verfügung steht.
- Eine technische Machbarkeit muss mit einem Prototypen getestet oder anhand eines Referenzprojektes nachgewiesen werden.



**Abbildung 5.3.** WfBI-Workflow-Architektur

In der Abbildung 5.3 bildet das WfMS die zentrale Komponente der Lösung. Die Workflow-Engine kennt alle Aktionen des Geschäftsprozesses und arbeitet diese entsprechend ab. Wie in der vorher beschriebenen Variante 1 werden die Anwendungen über Adapter angebunden, nur mit dem Unterschied, dass diese direkt von dem WfMS zur Verfügung gestellt und angesprochen werden.

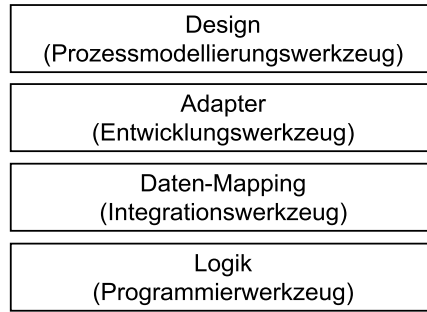
Es ist aber zu beachten, dass nicht alle WfMS-Produkte für eine solche Variante geeignet sind. Bei der Auswahl eines Produktes ist ein wichtiges Entscheidungskriterium, dass das WfMS-Produkt kein in sich geschlossenes System darstellt, sondern aufgrund seiner Architektur eine Anbindung auf einer tiefen technischen Ebene zulässt. Eine einfache funktionale Schnittstelle (API) deckt in der Regel die Anforderung einer performanten Anbindung nicht ab. Es muss daher die Möglichkeit bestehen, die Anbindung direkt am Kern der Workflow-Engine durchzuführen. Möchte man beispielsweise eine J2EE-basierte Workflow-Engine verwenden, so sollte man die Anbindung direkt im selben EJB-Container oder über JMS durchführen können. Als Produktbeispiel kann an dieser Stelle das in dem Kapitel Werkzeuge aufgeführte Produkt CARNOT genannt werden. Eine weitere technologische Möglichkeit ist die Verwendung von CORBA. Basiert ein WfMS-Produkt auf CORBA oder stellt es eine CORBA-Schnittstelle zur Verfügung, so kann die Anbindung der Anwendungen über entsprechende CORBA-Adapter erfolgen.

Bei allen Varianten muss bedacht werden, dass nicht nur das EAI-Produkt und das WfMS über Adapter angebunden werden, sondern auch die bestehenden Anwendungen. Je nach verwendeter Technologie und Architektur der Anwendungen können bestimmte Anbindungsarten bereits ausgeschlossen sein. Dies bedeutet, dass zuerst eine Analyse der zu integrierenden Anwendungen durchgeführt werden muss. Erst danach kann eine Auswahl der Adaptertechnologie erfolgen, die den Aufwand der Realisierung überschaubar hält. In Tabelle 5.1 sind die Kriterien aufgeführt, die bei der Analyse zu beachten sind.

**Tabelle 5.1.** Kriterien

Kriterium	Beschreibung
Betriebssystem	Auf welchem Betriebssystem wird die Anwendung betrieben? Dies kann entscheidend sein, da z.B. der DCOM-Betrieb nicht ohne weiteres auf einer UNIX-Plattform möglich ist.
Art der Schnittstelle	Welche Schnittstelle wird durch den Anwendungshersteller angeboten? Liegt eine einfache Datenschnittstelle oder eine Message-basierte Schnittstelle vor oder kann mit einer funktionalen Schnittstelle direkt auf die Anwendung zugegriffen werden?
Architektur	Welche Architektur wurde verwendet: liegt eine Mehrschichtenarchitektur vor, ein Fat-Client oder eine Host-Anwendung?
Technologie	Welche Technologie (z.B. CORBA, J2EE, .NET usw.) wird verwendet? Je nach Art der Technologie können Aussagen über den Realisierungsaufwand der Adapter getroffen werden.
Programmiersprache	Welche Programmiersprachen unterstützen die Produkthersteller für die Anbindung ihrer Anwendungen?

Diese Kriterien sind für alle Anwendungen zu analysieren, die an das WfMS gebunden werden sollen. Es kann so eine fundierte Entscheidungsgrundlage für die Auswahl der Technologie getroffen werden, mit der die Adapter realisiert werden sollen. In Abbildung 5.4 sind die Schichten einer Entwicklungsumgebung idealisiert dargestellt, die das Ergebnis einer solchen Analyse sein kann. In der obersten Schicht befindet sich das Werkzeug, mit dem die Geschäftsprozesse modelliert werden. Dieses Modellierungswerkzeug „kennt“ die Adapter, die die Anwendungen mit den Aktivitäten der Prozesse verbinden. Wird ein Adapter nicht durch den Produkthersteller selbst angeboten, so kann er durch eine Eigenentwicklung realisiert werden. Voraussetzung ist, dass die anzubindende Anwendung eine programmierbare Schnittstelle aufweist. Dazu benötigt man abhängig von der Programmiersprache und dem zu realisierenden Adapter eine entsprechende Entwicklungsumgebung.



**Abbildung 5.4.** Entwicklungswerkzeuge

In Sonderfällen kann es für die Realisierung einer Anbindung notwendig sein, die Daten einer Anwendung zu konvertieren. Dies kann durch ETL-Werkzeuge (vgl. Kapitel 2 Enterprise Application Integration (EAI)) effektiv realisiert werden.

Als weitere Variante kann die Integration der Workflow-Technologie in Client-/Server-Produkten genannt werden, d.h. in ein bestehendes Produkt wird das WfMS-Produkt integriert. Der Anwender behält in diesem Fall weitestgehend seine bisherige Oberfläche und wird durch ein WfMS innerhalb der Anwendung „geführt“. Da dies in der Regel mit sehr hohem Aufwand verbunden ist, weil die wenigsten Produkte eine solche umfangreiche Schnittstelle anbieten, sollte ein Unternehmen von einer solchen Variante Abstand nehmen. Auch die fachliche Funktionalität der Produkte ist in der Regel nicht in einer solchen Schnittstelle offengelegt bzw. dokumentiert.

### 5.2.3 Client-Varianten

Die bereits genannten Integrationsvarianten können noch weiter untergliedert werden. Es besteht die Möglichkeit, den Prozessbeteiligten einen separaten Workflow-Client zur Verfügung zu stellen oder die Workflow-Funktionalität in einem bestehenden Client (z.B. E-Mail-Client) zu implementieren. Diese Varianten unterscheiden sich von der eigentlichen Integration der bisher beschriebenen Varianten nicht grundsätzlich, sie werden nur ergänzt: Variante 1 oder Variante 2 werden wie dargestellt realisiert. Der Anwender erhält auf seinem Arbeitsplatz eine neue Workflow-Oberfläche oder verwendet weiterhin eine bestehende Oberfläche, in der die Workflow-Funktionalitäten implementiert werden.

#### Eigenständige Workflow-Client-Oberfläche

Mit diesem Client können alle Aktivitäten bearbeitet und abgeschlossen werden, die für die Prozesse erforderlich sind. Der Anwender erkennt in diesem Fall nicht, mit welchen Anwendungen er jeweils arbeitet, da die Workflow-Dialoge sich auf die Aktivität beziehen und nicht auf die dahinter liegenden Anwendungen. Dies bedeutet für die technische Realisierung, dass die Workflow-Standard-Funktionalitäten wie „Prozess starten“, „Prozess speichern“, „Prozess abrechnen“ und „Aktivität bearbeiten“ bereits unabhängig von den definierten Prozessen und deren

Aktivitäten implementiert sind. Der Prozessablauf und die Interaktionen mit den dahinter liegenden Anwendungen sind nur dem WfMS bekannt. Dieses „beliefert“ den Client mit den Aktivitäten, welche für den Bearbeiter bzw. die Rolle modelliert wurden.

Vorteil:

- Der komplette Funktionsumfang des WfMS steht dem Entwickler zur Verfügung.
- Dialoge können ohne Einschränkung realisiert werden.
- einfache Realisierung eines browserbasierten Clients.

Nachteil:

- Der Anwender muss sich – allerdings nur einmalig – mit einem neuen Client vertraut machen.

### **Client mit E-Mail-Integration**

Eine weitere Client-Variante ist die Integration der Workflow-Funktionalität in einen bestehenden Client. Dies ist häufig ein E-Mail-Client, der um die Workflow-Funktionalität erweitert wird. Der Anwender erkennt dies nur anhand zusätzlicher Menüeinträge oder durch einen zusätzlichen Ast in seiner Ordnerstruktur.

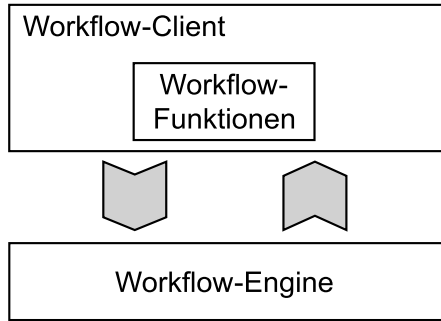
Vorteil:

- Der Anwender kann einen bestehenden Client verwenden und muss sich nur mit den neuen Workflow-Funktionen und nicht einer neuen Benutzeroberfläche vertraut machen.

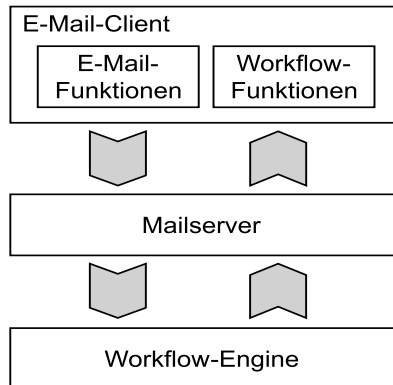
Nachteil:

- Nicht alle WfMS-Produkte bieten eine solche Integration an.
- Die Funktionalitäten können durch das E-Mail-Produkt eingeschränkt sein.
- Je nach E-Mail-Produkt läuft die Kommunikation zunächst über den E-Mail-Server und erst dann zur Workflow-Engine. Daraus folgt eventuell ein Performanzverlust und ein höherer Integrationsaufwand.
- Die Dialoge müssen der Technologie des E-Mail-Produktes entsprechen.
- Ein Web-Client ist nur eingeschränkt bzw. mit höherem Aufwand gegenüber dem Workflow-Client möglich.
- Im E-Mail-Client findet eine Vermischung von prozessorientierten Vorgängen und nicht prozessorientierten E-Mails statt, was eine prioritätsgesteuerte Bearbeitung erschwert.
- E-Mail-Server sind weiterhin bevorzugte Ziele von Viren- und Sabotageprogrammen – der Ausfall des E-Mail-Systems hätte auch den Ausfall der (ggf. unternehmensweiten) prozessorientierten Sachbearbeitung zur Folge.

In Abbildung 5.5 und Abbildung 5.6 werden die Architekturen dieser beiden Client-Varianten dargestellt. In der Praxis ist auch ein Mix aus beiden Varianten innerhalb der Prozesse realisierbar, d.h. ein Prozess wird über einen Internet-Auftritt eines Unternehmens gestartet und über die Workflow-Clients weitergeführt, welche entweder in eigenständigen Oberflächen realisiert oder in E-Mail-Clients integriert sind.



**Abbildung 5.5.** Workflow-Client



**Abbildung 5.6.** E-Mail-Client

#### 5.2.4 Authentifizierung und Autorisierung

Für die Authentifizierung und Autorisierung der Anwender sind neben der Authentifizierung gegenüber dem WfMS auch die angebotenen Anwendungen zu berücksichtigen. Es ist für den Bearbeiter des Prozesses wichtig, sich nicht an jeder benötigten Anwendung erneut anmelden zu müssen. Die Anmeldung muss durch das WfMS bzw. die Lösung erfolgen. Es muss während der Konzeption der Realisierung berücksichtigt werden, dass die Zugangsdaten für die benötigten Anwendungen vorliegen. Dies setzt voraus, dass das WfMS-Produkt und die angebotenen Anwendungen zum einen eine zentrale Benutzer-, Rollen- und Rechteverwaltung beinhalten und zum anderen diese Funktionalitäten in Form einer

dokumentierten Schnittstelle nach „außen“ geben. In großen IT-Lösungen, die viele anzubindende Anwendungen und eine hohe Anzahl von Anwendern beinhalten, empfiehlt es sich, einen LDAP-Server einzuplanen. Mit diesem kann eine zentrale Benutzerverwaltung mit zumindest den Benutzerkennungen, Passwörtern und dazugehörigen Rechten realisiert werden. Ist ein LDAP-Server implementiert, kann für Variante 2 beispielhaft folgendes Szenario nach der Eingabe der Benutzerkennung und des Passwortes gelten:

1. Die Benutzerkennung und das Passwort werden an das WfMS über den Workflow-Client weitergeleitet.
2. Kontrolle von UN und PW durch das WfMS und temporäre Speicherung der Daten, falls die Prüfung korrekt war.
3. Anfrage des WfMS an den LDAP-Server, welche Rechte für den Anwender bezüglich der Prozesssteuerung eingetragen sind.
4. Das WfMS stellt dem Anwender anhand der eingetragenen Rechte die Funktionalität für die Prozesssteuerung bereit.
5. Verwendet der Anwender in einem Prozess eine Aktivität, welche durch eine angebundene Anwendung realisiert ist, fragt im Hintergrund das WfMS mit den temporär gespeicherten Anwenderdaten bei dem LDAP-Server an, ob der Anwender berechtigt ist, die modellierte Funktion in der angebundenen Anwendung auszuführen. Ist dies der Fall, so wird der Anwender im Hintergrund durch das WfMS an der Anwendung angemeldet und kann die Funktion ausführen.

Dieser beschriebene Ablauf ist abhängig von dem WfMS und der Art der Realisierung. Er zeigt, dass durch die Integration eines LDAP-Servers der Anwender sich nur einmal an das WfMS anmelden muss und trotzdem die sichere Verwendung der Anwendungen ermöglicht wird. Dieses Verfahren nennt man in IT-Lösungen Single-Sign-On. Es kann auf unterschiedliche Art und Weise realisiert werden.

### 5.3 Kommentar

Es wurde gezeigt, dass beim Ansatz der prozessbasierten Integration der Umfang der Integration durch die technischen Anforderungen der Geschäftsprozesse bestimmt wird. Für die Durchführung eines solchen Ansatzes sind unterschiedliche Szenarien vorstellbar. Diese reichen von der „grünen Wiese“ bis zum bereits implementierten Integrationsserver.

Wichtig für eine effektive Durchführung dieses Ansatzes ist die Übereinstimmung der technischen Anforderungen folgender Komponenten:

- des verwendeten Modellierungswerkzeuges
- des WfMS-Produktes
- der eventuell verwendeten Integrationsserver sowie
- die Anbindungsfähigkeit der verwendeten Anwendungen über Adapter



Es gibt eine sehr große Vielfalt an auf dem Markt befindlichen Lösungskomponenten. Es bedarf daher einer genauen Evaluation dieser Produkte. Da wie erwähnt die Ausgangssituationen der Unternehmen unterschiedlich und die Anzahl der Produkte sehr groß sind, kann kein konkretes Produkt als optimale Auswahl für eine Geschäftsprozesslösung vorgeschlagen werden. Die im Kapitel Werkzeuge genannten Produkte sollen auch keinen kompletten Überblick über die Produkte geben. Die Auswahl soll vielmehr den Leser sensibilisieren, die wichtigsten Auswahlkriterien zu beachten, um selber eine richtige Einschätzung der Produkte vornehmen zu können.

In den vorangegangenen Kapiteln hat der Leser einen Einblick in die Technik und die Modellierung gewinnen können. Diese beiden Bereiche wurden im Kapitel Workflow-based Integration (Wfbl) zusammengefasst. Um die Umsetzung eines Wfbl-Projektes effektiv und effizient durchführen zu können, benötigt man neben der Werkzeugunterstützung (vgl. Kapitel 8 Werkzeuge) auch ein spezifisches Vorgehensmodell. Dieses wird im Kapitel Workflow-Projekte beschrieben.

## **6 Workflow-Projekte**

### **6.1 Einleitung**

Dieses Kapitel beschreibt eine Vorgehensweise für ein Integrationsprojekt in Verbindung mit einem Workflow (Workflow-Projekt) und die dafür benötigten Rollen der Projektbeteiligten. Es wird darauf hingewiesen, dass es sich hierbei um eine ideale Darstellung handelt und diese für den Praxisgebrauch an das jeweilige Unternehmen und Umfeld angepasst werden muss. Entscheidenden Einfluss auf den Umfang eines Projektes hat die Unternehmensgröße und die vorhandene IT-Infrastruktur des Unternehmens. Sind Applikationen und/oder Produkte bereits strategisch gesetzt, so beeinflussen diese Randbedingungen den Aufwand für das Projekt erheblich. Außerdem haben solche zuvor getroffenen Entscheidungen einen wichtigen Einfluss auf eventuelle Evaluierungen von weiteren Teillösungen für die technische Unterstützung von Geschäftsprozessen.

Es wird in diesem Kapitel aber nicht auf grundlegende Aspekte von IT-Projekten wie z.B. eine Risikoanalyse oder Fallback-Strategien eingegangen, da eine detaillierte Beschreibung derselben den Rahmen dieses Buches sprengen würde. Allerdings werden diese wichtigen Projektbestandteile häufig nicht ausreichend berücksichtigt. Dies zeigt sich in vielen IT-Projekten, in denen z.B. Risikoanalysen zu selten durchgeführt werden. Dadurch kommt es zu unvorhergesehenen Problemen, die vermieden oder zumindest verringert werden könnten.

### **6.2 Projektablauf/Vorgehensmodell**

In den vorangegangenen Kapiteln wurden die unterschiedlichen Ausgangssituationen für ein Workflow-Projekt erwähnt. Es gilt nun, diese auf die im Unternehmen tatsächlich vorhandenen Strukturen abzubilden. Durch solch eine „Standortbestimmung“ kann die Firmenleitung bzw. der Abteilungsleiter den fachlich richtigen Projektstart bestimmen und so die ersten Schritte für das Projekt definieren. Das Ergebnis einer solchen Standortbestimmung kann sein, dass erst die Geschäftsprozesse global dokumentiert und darauf folgend (Teil-)Geschäftsprozesse modelliert und realisiert werden. In dem folgenden Kapitel wird eine Vorgehensweise vorgestellt, deren Struktur es ermöglicht, ein Workflow-Projekt effektiv durchzuführen.

Zu Beginn muss jedoch definiert werden, was unter der Projektlaufzeit verstanden wird. Nach der Meinung des Autors wird in bzw. vor einem Projekt die globale

Sichtweise eines Projektes zu wenig berücksichtigt. Es kommt immer wieder vor, dass verzögerte Projekte gestoppt werden und das Ergebnis einer Prüfung (Review) dann zeigt, dass die Ursache für den schlechten Zustand des Projektes die begrenzte Sicht auf das Projektumfeld war. Ein Resultat eines solchen Reviews ist oft, dass kein Informationsaustausch und/oder keine Abstimmungen mit laufenden „Nachbarprojekten“ stattgefunden hat. Es ist daher wichtig, dass das eigene Projekt fest in die bestehende System- und Projektlanschaft integriert wird.

Workflow-Projekte haben hier einen entscheidenden Vorteil. Sie werden in der Regel von einer Fachabteilung ins Leben gerufen, so dass zumindest bei den fachlichen Anforderungen von einer Absprache innerhalb der Abteilung ausgegangen werden kann<sup>6</sup>. Somit muss man sich hauptsächlich bezüglich der technischen Realisierung mit anderen Projekten absprechen. Eine Ausnahme bilden hier die Prozesse in den IT-Abteilungen.

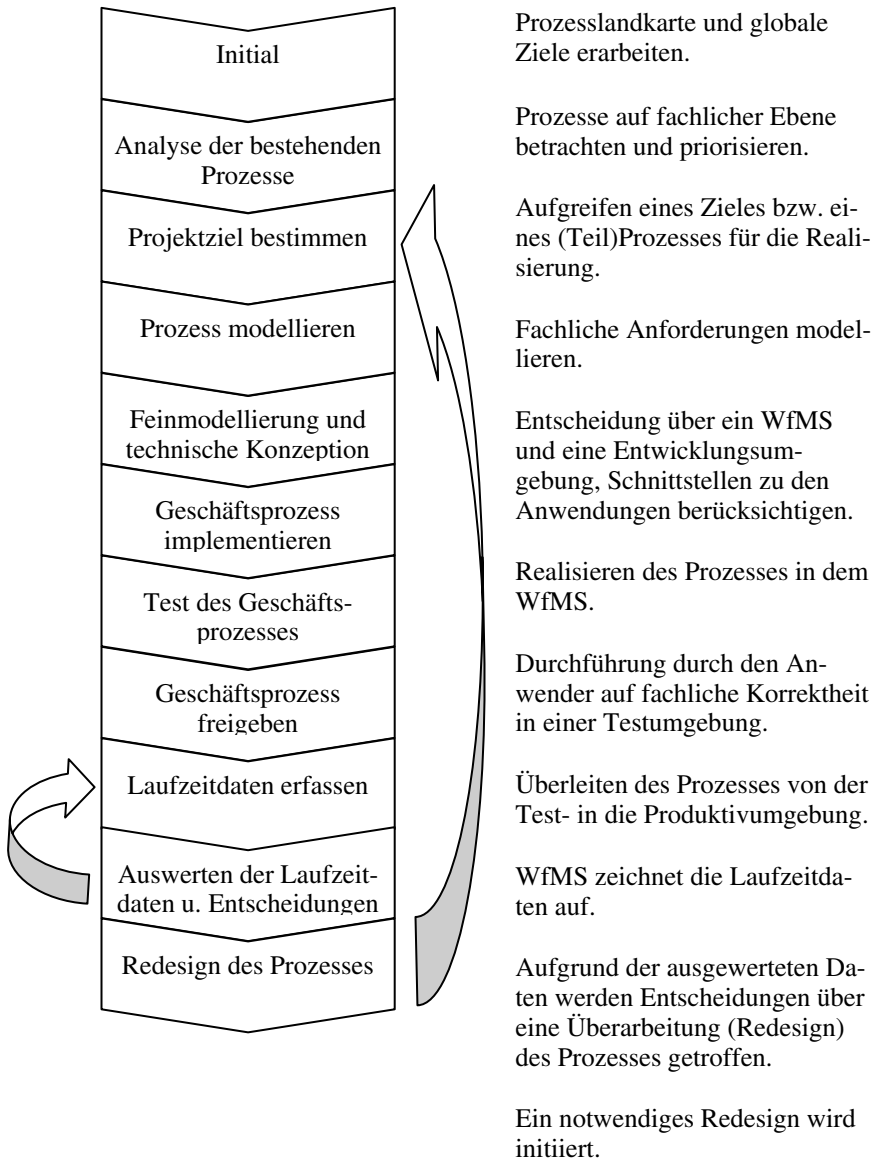
Fachabteilungen positionieren und starten ein Workflow-Projekt. Eine Fachabteilung ist somit der Auftraggeber und muss die Randbedingungen wie Budget und Anforderungen festlegen.

Der Auftraggeber muss sich im Klaren darüber sein, was die Projektlaufzeit für ihn darstellt und welche Aufgaben auf ihn und die Projektbeteiligten zukommen. Die Projektlaufzeit beginnt mit einem fertig formulierten Projektauftrag und endet – im Falle eines Workflow-Projektes –, wenn der Prozess oder die eingeführte Lösung nicht mehr verwendet wird. Demnach gehört zur Projektlaufzeit auch der Betrieb und die sich daraus ergebende Wartung eines Prozesses und der unterstützenden Systeme. Ist die Realisierung des Prozesses abgeschlossen und die IT-Lösung in Betrieb genommen worden, so wird das Projekt aus der Projektorganisation in die Linienorganisation überführt und die Wartung des Prozesses wird in der Abteilung durchgeführt, der der Prozessverantwortliche bzw. Auftraggeber zugeordnet ist. Dies schließt nicht aus, dass bei Erweiterungen wieder andere Abteilungen, insbesondere der IT-Bereich, involviert werden.

Die Wartung eines Prozesses kann ein erheblicher Kostenfaktor sein. Unter der Wartung versteht man nicht nur das Aufrechterhalten des Betriebes, also Ressourcen und CPU-Zeit, sondern auch Lizenzgebühren und Veränderungen eines Prozesses. Im Folgenden werden die einzelnen Phasen eines Workflow-Projektes dargestellt. Es handelt sich während der Laufzeit um insgesamt 11 Projektphasen, die teilweise iterativ durchlaufen werden (vgl. Abbildung 6.1). Aus diesen setzt sich das Gesamtprojekt zusammen.

---

<sup>6</sup> Das ist aber nicht immer der Fall. Es passiert bei Workflow-Projekten immer wieder, dass auch die Arbeitsabläufe innerhalb einer Organisationseinheit zuerst definitiv festgelegt werden müssen. Häufig haben verschiedene Mitarbeiter unterschiedliche Arbeitsweisen, die dann aufeinander abgestimmt werden müssen.



**Abbildung 6.1.** Übersicht der Projektphasen

Diese 11 Phasen werden in den nächsten Unterkapiteln chronologisch dem Projektverlauf folgend beschrieben. Zu den meisten Phasen findet der Leser im Kapitel Template Vorlagen, die er für sein Projekt verwenden kann. Zusätzlich stehen die Templates unter [www.workflow-based-integration.de](http://www.workflow-based-integration.de) zum Download bereit.

Die Templates haben keinen Anspruch auf Vollständigkeit, da auch sie an das entsprechende Umfeld angepasst werden müssen. Sie lösen aber die elementaren Fragen und Punkte aus, um eine Phase erfolgreich zu beenden.

Es wird noch einmal darauf hingewiesen, dass die gewählte Vorgehensweise keinesfalls sicherstellen kann, dass das dargestellte Vorgehensmodell auf alle Workflow-Projekte abgestimmt ist und alle möglichen Gegebenheiten abbildet. Auch die Detaillierung des Vorgehensmodells lässt einen sehr großen Spielraum zur Ausgestaltung der einzelnen Phasen zu.

### **6.2.1 Initial**

Wie in jedem Projekt müssen vor dessen Start die Anfangsbedingungen geklärt sein. Die Motivation für die Durchführung eines Workflow-Projektes kann unterschiedlich ausfallen. Aus dieser Motivation heraus wird aber die Entscheidung getroffen, welche Ziele durch das aktuelle Projekt erreicht werden sollen. Mögliche Ziele können z.B. sein, mehr Geschäftsvorfälle in kürzerer Zeit zu bearbeiten, die Qualität der Prozessbearbeitung zu steigern oder die Prozessbearbeitung (und damit auch das jeweilige Kundenbild) transparenter darzustellen. Dazu ist dann ein WfMS in Verbindung mit der bestehenden Middleware/Integrationsserver und/oder anderen Software-Lösungen in die Applikationswelt zu integrieren, oder in einem bereits implementierten WfMS sind weitere Prozesse zu implementieren.

Alle Anforderungen an das Projekt müssen als Ziele ausformuliert werden. Einen Überblick über mögliche Ziele erhält man, indem diese in einer tabellarischen Aufstellung zusammengetragen und mit den betroffenen Abteilungen und Mitarbeitern diskutiert werden. Um eine hohe Akzeptanz der Ziele zu erreichen, muss eine solche Diskussion mit allen Entscheidungsträgern und Betroffenen – auch der benachbarten Fachabteilungen – geführt werden. Es folgt der Entwurf einer Aufstellung, welche im Anhang als Vorlage unter T-Wfbl-01 (vgl. Kapitel 9 Template) zu finden ist.

Die ausformulierten Ziele müssen der Geschäftsleitung zur Genehmigung vorgelegt werden. Ist eine Einigung zwischen der Geschäftsleitung und der Fachabteilung erzielt worden, müssen die Verantwortlichen für die einzelnen Rollen bestimmt werden. Soll zum Beispiel ein neuer Prozess implementiert werden, so ist für diesen ein Prozessverantwortlicher zu bestimmen. Der Prozessverantwortliche kann parallel die Rolle des Projektleiters übernehmen, da diese Funktion über die komplette Projektlaufzeit durch eine Person wahrgenommen werden muss. Zusammen mit dem Auftraggeber, dem Prozessverantwortlichen, dem Projektleiter und ggf. der Geschäftsleitung werden der Geschäftsprozess und die Prozessziele festgelegt und dokumentiert.

Unabhängig vom Projektziel ist es erstrebenswert, eine Prozesslandkarte anzufertigen, welche eine globale Darstellung der betroffenen Prozesse ermöglicht. In einer Prozesslandkarte (T-Wfbl-02) wird jeder Geschäftsprozess mit einem einfachen Symbol dargestellt. Zudem beschreibt die Prozesslandkarte alle Daten- und

Materialflüsse zwischen den Geschäftsprozessen. Durch Einbindung der betroffenen Fachabteilungen können somit die Kerngeschäftsprozesse mit ihren Hilfsprozessen sehr schnell grafisch dargestellt werden. Man gewinnt einen Überblick, wie die Geschäftsprozesse zusammenspielen, und zusätzlich hilft es dabei, einzelne zusätzliche Geschäftsprozesse zu identifizieren. Zu einem späteren Zeitpunkt werden die Geschäftsprozesse in weitere Teilprozesse und Schritte zerlegt und detailliert dokumentiert.

Nach der Erstellung einer solchen Übersicht werden die betroffenen Prozesse nach ihrer Bedeutsamkeit für das Kerngeschäft beurteilt. Man wählt hier am besten den pragmatischen Ansatz und analysiert im Unternehmen die Prozesse nach den folgenden Kriterien:

- Einfluss des Prozesses auf das Unternehmensziel
- Beurteilung des Kosten/Nutzenverhältnisses
- Steigerung der Qualität und der Zufriedenheit der Prozessbeteiligten (Kunden, Mitarbeiter, ...)
- Unterstützung der Unternehmensziele
- Eignung des Prozesses in der Umsetzung für die Einführung eines WfMS

Im Anhang werden Vorlagen bzw. Templates unter T-Wfbl-03 und T-Wfbl-04 zur Verfügung gestellt.

Falls ein WfMS bereits implementiert ist und das Ziel definiert wird, weitere Prozesse damit zu unterstützen oder bereits unterstützte Prozesse zu optimieren z.B. durch die Anbindung weiterer Applikationen, dann sollte in erster Linie das Kosten-/Nutzenverhältnis und die Steigerung der Qualität als Entscheidungsgrundlage herangezogen werden. Wichtig bei dieser ersten Phase ist es, sich von einer Abteilungs- bzw. eingeschränkten Sichtweise zu lösen. Dies erreicht man durch die oben erwähnte Prozesslandkarte, welche mindestens die Prozesse enthalten muss, die für das Erreichen des globalen Zieles als notwendig angesehen werden.

Das Ergebnis der Initial-Phase muss die Prozesslandkarte und die ausformulierten Ziele beinhalten, welche mit der Geschäftsleitung abgestimmt wurden. Außerdem müssen die beteiligten Personen für die im Kapitel Projektorganisation und Projektrollen genannten Rollen bestimmt werden (vgl. T-Wfbl-05). Bei der Einführung eines WfMS wird nicht nur die eigene Abteilung oder das eigene Unternehmen betrachtet, sondern auch benachbarte Fachabteilungen oder sogar angebundene Lieferanten, die an dem Geschäftsprozess beteiligt sind. Auf Grund der Globalisierung der Märkte ist es wichtig, den Geschäftsprozess von Anfang bis Ende zu betrachten und danach die Teile zu analysieren, deren Realisierung oder Optimierung für die Unternehmensziele wichtig und einflussreich ist. Diese globale Sicht betrifft die fachliche Prozessabwicklung, aber auch die spätere technische Realisierung und somit die IT-Abteilung.

6.2.2 Analyse der bestehenden Prozesse

Wurde ein Überblick anhand der Prozesslandkarte gewonnen und sind die globalen Ziele bestimmt und im Unternehmen veröffentlicht worden, dann müssen jetzt die bestehenden Geschäftsprozesse analysiert werden. Es wird mit jenen Geschäftsprozessen begonnen, die nach dem oben beschriebenen Verfahren identifiziert wurden.

Für jeden zu analysierenden Geschäftsprozess werden die Zulieferprozesse<sup>7</sup> und deren Aktivitäten der Zulieferprozesse tabellarisch aufgelistet und die Messkriterien bestimmt. Diese Messkriterien können z.B. Liegezeiten, Transportzeiten, Bearbeitungszeiten und/oder Durchlaufzeiten sein. Aber auch die Kosten und die Fehlerhäufigkeit bzw. Qualität, die der Geschäftsprozess zur Folge hat, sowie die Ressourcen, die er nutzt, sollten berücksichtigt werden. In der unten dargestellten Checkliste (vgl. Tabelle 6.1) sind Messkriterien für die Analyse der Geschäftsprozesse aufgeführt. Mit diesen Kriterien wird die aktuelle Performance des Geschäftsprozesses bestimmt, die später als Basis für die Bewertung (qualitative Auswertung) des Geschäftsprozesses verwendet wird.

Für ein besseres Verständnis und eine genauere Übersicht des Geschäftsprozesses empfiehlt es sich, für jeden Prozess eine grafische Beschreibung zu erstellen. Diese soll dem Aufbau der Prozesslandkarte entsprechen und einen Überblick über den Verlauf und den Inhalt des Geschäftsprozesses inkl. seiner Prozessschritte geben. Es wird in dieser Darstellung der Daten- und Materialfluss zwischen den Schritten aufgezeigt. Der Vorteil einer solchen Übersicht ist, dass man im Vergleich zur textuellen Beschreibung einen schnelleren und besseren Überblick über den Geschäftsprozess erhält (vgl. Kapitel 4 Prozessmodellierung).

„Ein Bild sagt mehr als tausend Worte“

Tabelle 6.1. Messkriterien für Geschäftsprozesse

Ergebnis des Geschäftsprozesses
Anzahl der beteiligten Personen
Änderungen pro Jahr an dem Geschäftsprozess
Häufigkeit des Geschäftsprozesses
Fehlerhäufigkeit
Durchlauf-, Liege-, Bearbeitungs- und Transportzeiten
Schnittstellen zu weiteren Geschäftsprozessen (In-/Output)
Kundenrelevanz
Umfang der Kosten für den Personaleinsatz
Umfang der Kosten für die Materialien
Umfang der Kosten für die IT-Infrastruktur

<sup>7</sup> Zulieferprozesse unterstützen die identifizierten Geschäftsprozesse bei deren erfolgreicher Durchführung.

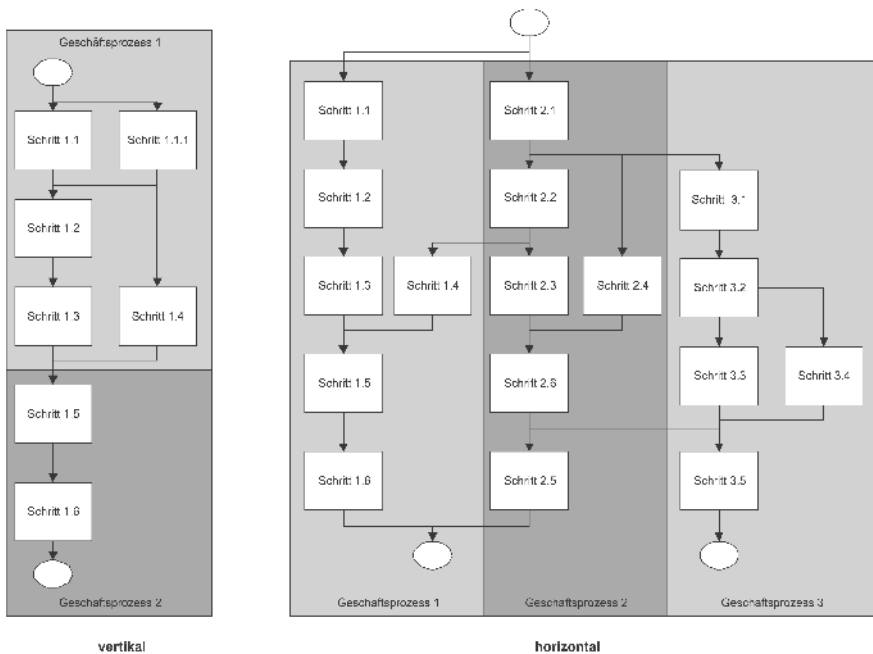
Für das Erfassen der Zulieferprozesse, der Mengengerüste und der Ressourcen stehen die drei Templates T-Wfbl-06, T-Wfbl-07 und T-Wfbl-08 zur Verfügung.

### 6.2.3 Projektziel bestimmen

Liegt die Dokumentation bzw. die Analyse der ausgewählten Geschäftsprozesse vor, dann kann das Projektziel bestimmt werden.

Um die Projektlaufzeit und den Aufwand für das Projekt in einem überschaubaren Rahmen zu halten, empfiehlt es sich, zunächst einen der zuvor analysierten Geschäftsprozesse für die Realisierung auszuwählen. Diese Selektion kann etwas schwierig sein, da ein Geschäftsprozess gefunden werden muss, der aus der Prozesslandschaft weitgehend herausgelöst werden kann. Anhaltspunkte liefert die zuvor erstellte Analyse der fachlichen Schnittstellenbeschreibung zu anderen Geschäftsprozessen.

Bei der Entscheidung für einen Geschäftsprozess wird der Umfang der horizontalen und der vertikalen Größe des Geschäftsprozesses berücksichtigt. Die horizontale Größe eines Geschäftsprozesses ist z.B. die Anzahl der Abteilungen, die zum selben Zeitpunkt parallel an unterschiedlichen Schritten des Geschäftsprozesses arbeiten. Die vertikale Größe ist die Anzahl der Aktivitäten des Geschäftsprozesses innerhalb einer Abteilung sowie abteilungsübergreifend (vgl. Abbildung 6.2).



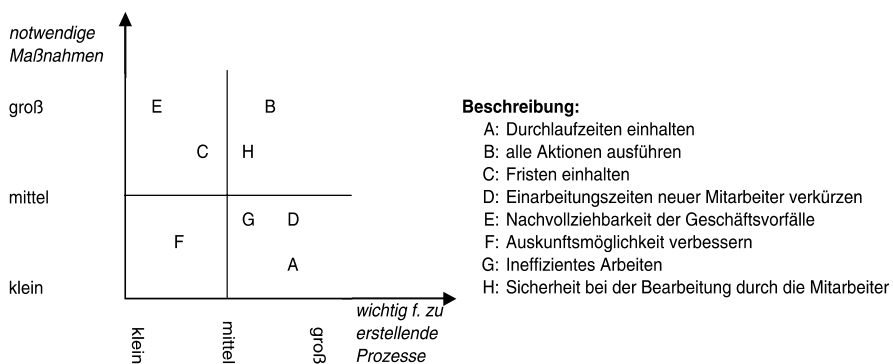
**Abbildung 6.2.** Horizontale und vertikale Größe von Geschäftsprozessen



Da die Erfahrung gezeigt hat, dass es nicht empfehlenswert ist, eine firmenweite Restrukturierung mit der Einführung eines WfMS zu kombinieren, wählt man einen Geschäftsprozess, welchen man aus der Prozesslandschaft herauslösen kann. Idealerweise ist es ein Geschäftsprozess, in dem die vertikalen und horizontalen Schnittstellen zu anderen Geschäftsprozessen relativ locker definiert sind, d.h. die Schnittstellen bekommen in der ersten Realisierungsphase keine technische Unterstützung und werden organisatorisch geregelt. Der daraus resultierende Medienbruch wird im ersten Schritt des Gesamtprojektes in Kauf genommen (z.B. durch Ausdrucken einer Arbeitsanweisung etc.). Es wird so die spätere Modellierung vereinfacht und eine Restrukturierung – falls erforderlich – innerhalb des Geschäftsprozesses erleichtert. Neben der lockeren Anbindung ist auch darauf zu achten, dass die Anzahl der Schnittstellen gering gehalten wird. Eine geringe Anzahl der Schnittstellen wirkt sich später positiv auf die Möglichkeit der Modularisierung und den Aufwand der Implementierung des Geschäftsprozesses in einem WfMS aus.

Sind die oberen Schritte durchgeführt und wurde ein geeigneter Geschäftsprozess gefunden, der auch für den Einsatz eines WfMS geeignet ist, gilt es nun, die Ziele für das Projekt detailliert zu bestimmen. In einem Workflow-Projekt, durch das ein WfMS eingeführt werden soll, können die Ziele sehr unterschiedlich und zahlreich sein. Aus diesem Grund fasst man die Ziele zusammen („clustern“). Hierfür wird ein Zielfortfolio (Teufel et al. 1995) verwendet, in dem die zuvor bestimmten Ziele eingetragen werden.

Zuerst aber werden diese Ziele definiert. Dabei muss immer eine Kontrolle stattfinden, ob die definierten Ziele des Workflow-Projektes die zuvor definierten Unternehmensziele unterstützen. Diese Ziele werden erst qualitativ formuliert und in einem weiteren Schritt quantitativ bewertet. Dies ermöglicht später die Kontrolle, ob/wie weit die Ziele des Projektes erreicht wurden. Sind die Ziele des Projektes erfasst, werden diese in das Zielfortfolio eingetragen (vgl. Abbildung 6.3).



**Abbildung 6.3.** Beispiel eines Zielfortfolios

Für das Erfassen des Verbesserungspotentials, der Definition der Teilprojekte bzw. Meilensteine und der Aufnahme der Schnittstellen stehen die drei Templates T-WfbI-09, T-WfbI-10 und T-WfbI-11 zur Verfügung.

### 6.2.4 Prozess modellieren

Die Modellierung eines Geschäftsprozesses ist sehr stark mit der Organisation des Unternehmens verknüpft. Es muss bei der Modellierung des Geschäftsprozesses darauf geachtet werden, dass bei jeder Phase hinterfragt wird, ob der modellierte Geschäftsprozess mit der bestehenden Organisationsform optimal bearbeitet werden kann, oder ob es sinnvoller ist, die Organisation dem neuen Ablauf anzupassen. Im Folgenden werden die organisatorischen Möglichkeiten für die Geschäftsprozesse aufgezählt.

#### Arbeitsteilung

Werden Schritte sequentiell ausgeführt, so ist darauf zu achten, dass ein einzelner Mitarbeiter möglichst viele Schritte<sup>8</sup>/Aktivitäten bearbeitet. Gegebenenfalls muss der Mitarbeiter auf die neuen Schritte geschult werden. Mit solch einer objektbezogenen Arbeitsteilung kann die Transport- und Liegezeit innerhalb des Prozesses verkürzt werden.

Eine andere Art der Arbeitsteilung ist die verrichtungsorientierte Arbeitsteilung, d.h. jeder Schritt wird durch einen anderen Mitarbeiter durchgeführt. Dies hat zur Folge, dass die Transport- und Liegezeiten im Vergleich zur eigentlichen Bearbeitung relativ hoch sein können.

#### sequentiell/parallel

Nach Möglichkeit werden Schritte, welche nicht zwingend sequentiell bearbeitet werden müssen, parallel bearbeitet. Müssen Schritte sequentiell bearbeitet werden, kann durch eine objektbezogene Arbeitsteilung die Transport- und Liegezeit der Schritte verkürzt werden.

#### Ausbildung

Wird durch die Ausbildung der Mitarbeiter das Know-how erhöht, kann eine Verkürzung der Durchlaufzeit erreicht werden. Neben der fachlichen Ausbildung in die Tiefe (eigener Aufgabenbereich) ist auch eine Ausbildung in die Breite (Aufgabenbereiche anderer Abteilungen) möglich. Dadurch kann eine flexiblere Einsatzmöglichkeit der Mitarbeiter erreicht werden. Dies wirkt sich positiv bei saisonalen Geschäftsprozessen aus, da Mitarbeiter aus anderen Bereichen zum Ausgleich von Arbeitsspitzen herangezogen werden können. Durch eine bessere Ausbildung kann auch die Anzahl der Irläufer minimiert werden und somit neben der Verkürzung der Gesamtdurchlaufzeit des Geschäftsprozesses auch die Kundenzufriedenheit erhöht werden.

---

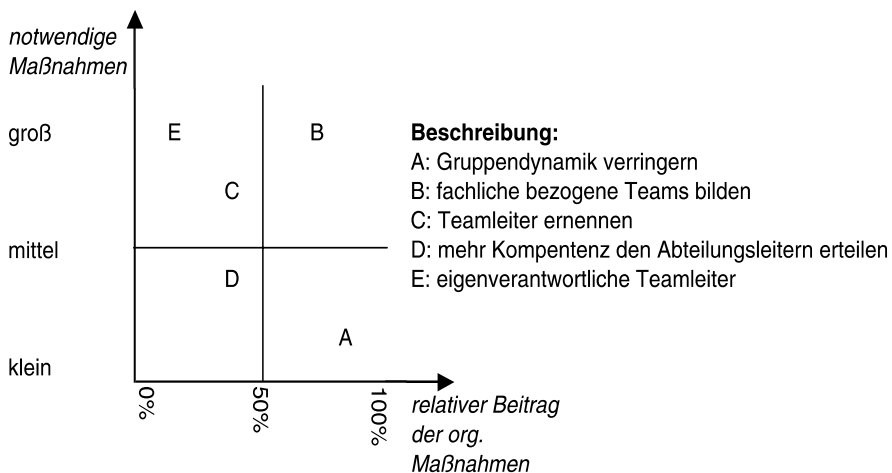
<sup>8</sup> In einem Schritt werden alle Aktivitäten, die an einem Arbeitsplatz bearbeitet werden können, zusammengefasst; d.h. unterschiedliche Schritte sind nur dann die Folge, wenn ein Arbeitsplatzwechsel zwingend vorgeschrieben ist oder auf externe Ereignisse zur Synchronisation gewartet wird.

### Entscheidungsbefugnisse

Durch die prozessbezogene dedizierte Delegation von Entscheidungen ist es möglich, die Arbeitsbelastung einzelner Mitarbeiter zu verringern und somit die Durchlaufzeit eines Geschäftsprozesses zu verkürzen. Wird z.B. bei einem Kreditantrag erst ab einer Kredithöhe von 10.000 € ein Vorgesetzter für die Genehmigung benötigt, so können die Kreditanträge mit einer geringeren Summe durch den Sachbearbeiter sofort genehmigt werden. Somit verteilt sich die Arbeitslast auf mehrere Mitarbeiter.

Zur Erinnerung: Es ist bei der Implementierung eines WfMS darauf zu achten, dass die Ablauflogik und die Benutzeroberflächen für die fachliche Logik und deren Bearbeitung strikt getrennt werden. Die Ablauflogik und die verwendeten Oberflächen haben bei allen Prozessen gleich zu funktionieren. Dies bedeutet, dass sich Mitarbeiter anderer Abteilungen bei Lastspitzen nur in die fachlichen Veränderungen einarbeiten müssen, die Prozesssteuerung ihnen aber schon vertraut ist.

Hat man einen Überblick über die organisatorischen Möglichkeiten erhalten, welche die Ziele positiv beeinflussen, dann werden diese Möglichkeiten in Verbindung mit den Zielen in das Zielportfolio eingetragen. Durch diese Art der Visualisierung kann man ablesen, welche Ziele durch organisatorische Änderungen erreicht werden können. Liegen nach der Analyse die meisten Ziele im oberen rechten Quadranten des Zielportfolios, so bedeutet das, dass sehr viele Ziele nur durch organisatorische Änderungen erreicht werden können. In diesem Fall sollte man die Einführung eines WfMS noch einmal überdenken und das Unternehmen oder den Bereich zunächst einer möglichen Restrukturierung unterziehen (Abbildung 6.4).



**Abbildung 6.4.** Ziele mit organisatorischen Maßnahmen

Bei diesem Verfahren besteht auch die Möglichkeit, technische Optimierungsmaßnahmen zu untersuchen. An dieser Stelle soll aber darauf nicht eingegangen werden, da die Gefahr besteht, sich bereits zu detailliert mit unterstützenden Soft-

ware-Produkten und deren Einsatz auseinander zu setzen. Somit würde relativ schnell das Überarbeiten der organisatorischen und fachlichen Optimierungsmöglichkeiten in den Hintergrund geraten.<sup>9</sup>

Sind die organisatorischen Maßnahmen bestimmt, dann beginnt die Modellierung des Geschäftsprozesses. Zuvor wird die Methode bestimmt, mit welcher die modellierten Geschäftsprozesse dokumentiert werden. Es sollte möglichst eine allgemeine Form der Darstellung gewählt werden, damit die spätere Umsetzung in dem technischen WfMS nicht durch eine spezielle Art der Darstellung mit erhöhtem Aufwand verbunden oder ggf. sogar ausgeschlossen ist.

Im Folgenden wird ein Überblick über einige mögliche Methoden und Darstellungsformen gegeben, welche im Kapitel Prozessmodellierung beschrieben wurden und keinen Anspruch auf Vollständigkeit haben.

- EPK (Ereignisgesteuerte Prozessketten)
- ICN (Information Control Nets)
- OMT (Object Modeling Technique)
- OSSAD (Office Support Systems Analysis and Design)
- PADM (Process Analysis and Design Method)
- Petrinetze
- RAD (Role Activity Diagrams)
- SOM (Semantische Objektmodelle)
- UML (Unified Modeling Language)

Die Entscheidung für eine Methode und eine Darstellungsform sollte sich an dem vorhandenen Know-how im Unternehmen, aber auch an dem Bekanntheitsgrad der Darstellungsform orientieren. Da diese ein entsprechendes Werkzeug (Tool) zur Modellierung benötigen, empfiehlt es sich, eine der gängigsten Methoden auszuwählen. Dies ist neben Petrinetzen und EPK die Unified Modeling Language (UML). Die UML hat ihren Ursprung in der objektorientierten Software-Entwicklung und ist nach Meinung des Autors eine Methode, die über den Bereich der Geschäftsprozessmodellierung hinaus verstanden wird. Mit der Entscheidung für ein Tool bzw. eine Methode empfiehlt es sich, keine einseitigen Abhängigkeiten zu einem Tool-Hersteller zu schaffen, indem eine Methode gewählt wird, die nur von dem jeweiligen Tool-Hersteller unterstützt wird. An dieser Stelle sei auf das Kapitel Prozessmodellierung verwiesen – hier werden die anderen beiden Methoden (EPK und Petrinetze) beschrieben. Die folgenden Beispiele werden mit UML für die Geschäftsprozessmodellierung dargestellt.

Sind die Entscheidungen über die Modellierungsmethoden und die Darstellungsart getroffen, kann mit der eigentlichen Grobmodellierung begonnen werden. Für die-

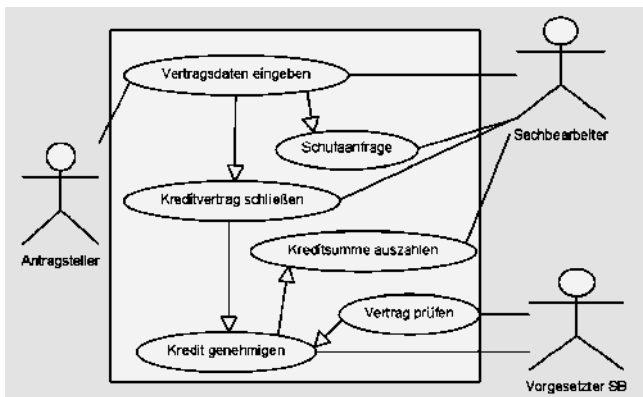
---

<sup>9</sup> Eine andere Ausgangssituation ist, wenn bereits ein WfMS eingeführt wurde und nun zusätzliche Geschäftsprozesse durch dieses WfMS unterstützt werden sollen. In diesem Fall ist die Analyse der technischen Optimierungsmöglichkeiten unter Berücksichtigung der IT-Gegebenheiten durchaus sinnvoll.

sen Zweck müssen alle Anwendungsfälle (Use Cases) für den ausgewählten Geschäftsprozess dargestellt werden. Die Anwendungsfälle sollten dabei in der Reihenfolge aufgeführt werden, in der sie später auch bearbeitet werden, also unter Beachtung des chronologischen und logischen Zusammenhangs zur Erreichung des Geschäftsprozesszieles. Hierbei werden eventuelle organisatorische Veränderungen noch nicht sofort berücksichtigt.

In der ersten Phase der Grobmodellierung konzentriert man sich auf das eigentliche Ziel des Geschäftsprozesses und lässt zunächst außer Acht, wer an der Erreichung des Zieles mitwirkt. Es ist wichtig, zunächst diesen Top-down-Entwurf zu wählen, damit man mit einem geringen Aufwand einen schnellen Überblick über den zu realisierenden Geschäftsprozess erhält.

Um den Detaillierungsgrad der Grobmodellierung zu veranschaulichen, wird an dieser Stelle ein Beispielgeschäftsprozess in UML modelliert. Dieser Geschäftsprozess wird in den folgenden Phasen weiter verfeinert (Abbildung 6.5).



**Abbildung 6.5.** Use Cases Beispielgeschäftsprozess „Kreditantrag“

Nach der groben Modellierung des Geschäftsprozesses können die Akteure bzw. die Rollen angegeben werden, die von innen und von außen<sup>10</sup> auf den Geschäftsprozess einwirken. Diese können aber auch erst während der Feinmodellierung bestimmt werden. Um einen Geschäftsprozess besser verstehen zu können, bietet sich aber eine „provisorische“ Zuweisung der Rollen an. Neben der grafischen Grobmodellierung (Beschreibung) wird der Geschäftsprozess auch noch verbal beschrieben. Die Reihenfolge für das Erstellen der Beschreibung (grafisch und textuell) ist frei wählbar. Es ist wichtig, dass beide Formen der Beschreibung erstellt werden, um so einen hohen Grad der Vollständigkeit zu erreichen und auch gegenseitig eine Vervollständigung der jeweiligen Darstellung zu ermöglichen. Im Folgenden Kasten wird eine mögliche verbale Beschreibung des Beispielprozesses aufgeführt.

<sup>10</sup> Mit „innen“ sind Mitarbeiter des Unternehmens und mit „außen“ sind Kunden, Lieferanten usw. gemeint.

In dem Kreditinstitut soll ein Kredit an einen Kunden vergeben werden. Der Antragsteller und der Sachbearbeiter erfassen die persönlichen Daten und die gewünschte Kreditsumme des Antragstellers. Nach erfolgter Datenerfassung soll eine Schufa-Anfrage erfolgen. Fällt diese Anfrage für den Kunden positiv aus, so kann der Antrag geschlossen und zur Genehmigung an den Vorgesetzten weitergegeben werden. Der Vorgesetzte des Sachbearbeiters muss die Möglichkeit haben, alle Vertragsdaten und Prozessdaten vor der Genehmigung einzusehen. Nach der Genehmigung des Kredites wird der Kreditvertrag von Antragsteller unterschrieben und die Kreditsumme an den Antragsteller ausgezahlt.

### 6.2.5 Feinmodellierung und technische Konzeption

Mit den Ergebnissen aus den zuvor durchgeführten Phasen wurde eine Basis geschaffen, um den Geschäftsprozess im Detail beschreiben zu können.

Das Ergebnis der Feinmodellierung ist wieder eine grafische und eine textuelle Beschreibung. Für die grafische Beschreibung wird nach dem UML-Anwendungsdiagramm (Use Case Diagram) nun ein Aktivitätsdiagramm (Activity Diagram) erstellt. In diesem Diagramm werden Ereignisse, Zustände und die Prozessaktivitäten dargestellt. Im Folgenden werden zuerst die drei Begriffe näher erläutert.

#### **Aktivität**

Ein Geschäftsprozess ist in einzelne Aktivitäten aufgeteilt, welche durch Ereignisse gestartet bzw. freigegeben werden. Laut der WfMC entspricht eine Aktivität einem Arbeitsschritt, auch Vorgangsschritt, Aufgabe oder Tätigkeit.

#### **Ereignis**

Ein Ereignis ist eine Zustandsänderung in einer Aktivität. Ereignisse können aus anderen Systemen und Geschäftsprozessen (extern) oder auch innerhalb des Geschäftsprozesses eintreffen oder ausgelöst werden.

#### **Zustand**

Der Geschäftsprozess ist immer in einem bestimmten Zustand. Der Zustand kann sich durch das Beenden einer Aktivität oder durch das Eintreffen eines (externen oder internen) Ereignisses ändern.

Ein Geschäftsprozess wird nach B. Oestereich (B. Oestereich 1995) durch folgende Eigenschaften beschrieben:

#### **Vorbedingungen**

Erwarteter Zustand des Systems, bevor der Anwendungsfall eintritt.

#### **Nachbedingung**

Erwarteter Zustand des Systems, nachdem der Anwendungsfall erfolgreich durchlaufen wurde.

**Nichtfunktionale Anforderungen**

Zusicherungen, die für Design und Realisierung wichtig sind, Plattform- und Umgebungsvoraussetzungen, qualitative Aussagen, Antwortzeitanforderungen, Häufigkeitsschätzungen, Entwicklungsprioritäten etc.

**Szenario-Beschreibung**

Beschreibung des Anwendungsfalles, ggf. gegliedert in nummerierte Einzelpunkte.

**Variationen**

Abweichungen und Ausnahmen zum Szenario und Beschreibung des alternativen Szenarios für diese Fälle.

**Regeln**

Geschäftsregeln, fachliche Abhängigkeiten, Gültigkeits- und Validierungsregeln usw., die im Rahmen des Szenarios von Bedeutung sind.

**Services**

Liste von Operationen und ggf. Objekten, die im Rahmen des Szenarios benötigt werden.

**Ansprechpartner, Sitzungen**

Liste der Personen, mit denen der Anwendungsfall erarbeitet bzw. durchgesprochen wurde, wann diese Sitzungen stattfanden etc.; ggf. mit welchen Personen noch zu sprechen ist, welche Rollen/Funktionen die Beteiligten einnehmen usw.

**Anmerkungen/offene Fragen**

Hier ist beispielsweise Platz zur Dokumentation von wichtigen fachlichen Entscheidungen (gelegentlich existieren verschiedene, fachlich konkurrierende Standpunkte zu einem Anwendungsfall).

**Dialogbeispiele oder –muster**

Beispieldialoge, Bildschirmkopien, Druck- und Formularbeispiele etc., die den Anwendungsfall veranschaulichen und die in den Gesprächen mit den Fachabteilungen etc. verwendet wurden.

**Diagramme**

Zum Beispiel Sequenz-, Aktivitäten- und Zustandsdiagramme, die das aus dem Anwendungsfall resultierende oder das hierfür notwendige interne Systemverhalten darstellen sowie systeminterne Abhängigkeiten und Zustandsänderungen im Zusammenhang mit diesem Anwendungsfall darstellen.

Ergänzend zu den Aspekten von B. Oestereich ist noch die zeitliche Betrachtung des Anwendungsfalles zu nennen. Es muss beschrieben werden, welche maximale Liegezeit vor der Bearbeitung, welche maximale Bearbeitungszeit für bestimmte Aktivitäten und welche maximale Durchlaufzeit des Anwendungsfalles zulässig sind. Diese Zeiten eines Geschäftsprozesses werden in den meisten WfMS als Fristen (Timer) bezeichnet und haben zur Folge, dass nach dem Ablauf solcher Fristen der modellierte Eskalationsprozess gestartet werden muss.

Sind die oben genannten Aspekte für jeden Anwendungsfall dokumentiert, dann kann anhand des Aktivitätsdiagramms (UML) der Geschäftsprozess grafisch dokumentiert werden. Hierfür wird wieder der zuvor grob modellierte Anwendungsfall verwendet. In Abbildung 6.6 werden neben den UML-Notationen Vorschläge von B. Oestereich für die Darstellungen von Fristen und Wiedervorlage verwendet.

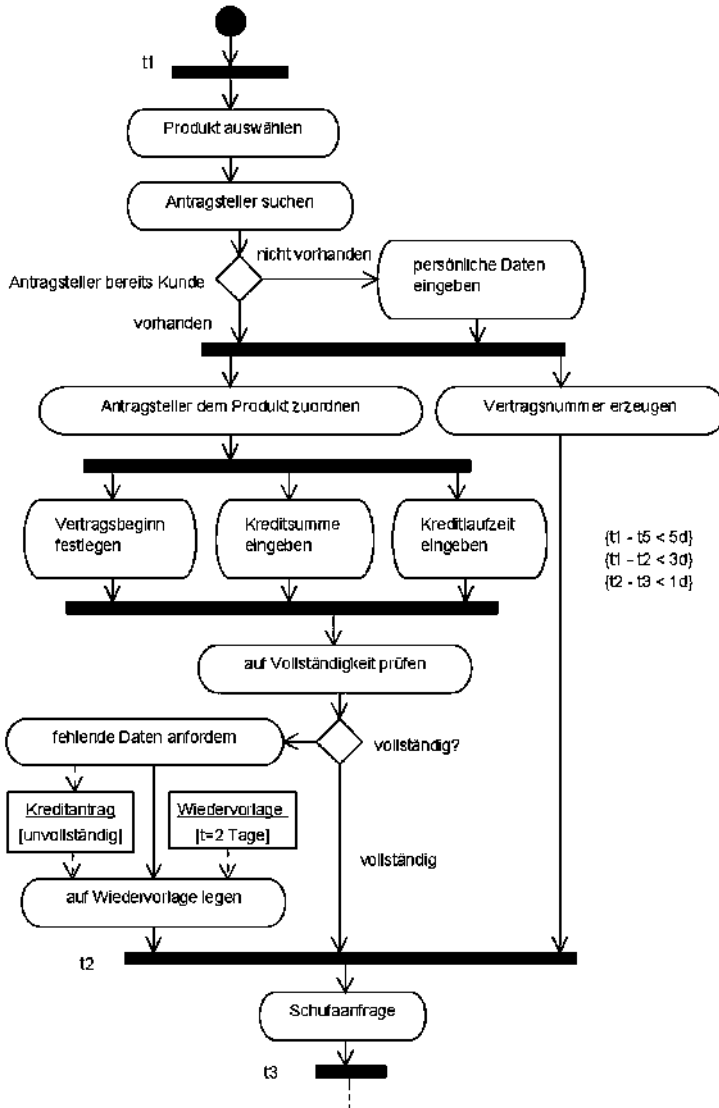


Abbildung 6.6. Aktivitätsdiagramm<sup>11</sup> für „Kreditantrag“ (Ausschnitt)

<sup>11</sup> t : time (Zeitpunkt) und 1d : steht für 1 Tag



Nachdem die fachlichen Anforderungen durch die Fachabteilungen und einem prozesserfahrenen Mitarbeiter mit einem hohen Detaillierungsgrad modelliert wurden, wird mit der technischen Konzeption begonnen.

Ein wichtiger Punkt ist die eventuell noch ausstehende Entscheidung über ein WfMS und eine Entwicklungsumgebung. Eine detaillierte Beschreibung einer Evaluierung für ein WfMS und einer passenden Entwicklungsumgebung würde den Rahmen dieses Kapitels und auch des Buches sprengen. Die folgenden Entscheidungshilfen geben nur einen Überblick über eine solche Evaluierung und haben keinen Anspruch auf Vollständigkeit. Eine weitere Hilfe für die Evaluierung eines WfMS ist im Kapitel Werkzeuge zu finden. Falls eine Evaluierung bereits erfolgt ist, kann man die beiden Abschnitte WfMS und Entwicklungsumgebung überspringen.

Um ein WfMS effektiv einsetzen zu können, ist es wichtig, die Schnittstellen der anzubindenden Applikationen zu analysieren. Im zuvor modellierten Geschäftsprozess wurden die Aktivitäten in den Arbeitsschritten festgelegt. Aufgrund dieser Aktivitäten und der daraus folgenden Funktionen erkennt man die notwendigen Applikationen bzw. Teillösungen in dem Geschäftsprozess. Um eine hohe Wiederverwendbarkeit der bereits im Unternehmen vorhandenen Applikationen und Teillösungen zu erreichen, empfiehlt es sich als erstes eine Bestandsaufnahme der derzeit eingesetzten Lösungen durchzuführen. Hierfür steht im Anhang das Formular T-WfBI-12 zur Verfügung. Erst nach dieser Recherche müssen die noch fehlenden Applikationen und Teillösungen evaluiert werden. Als Beispiel für die Analyse der Schnittstellen wird wieder der zuvor modellierte Prozess „Kreditantrag“ verwendet.

In Abbildung 6.7 wurde das Aktivitätsdiagramm dafür verwendet, die möglichen Schnittstellen zu den Fremdsystemen darzustellen. Ausgehend von der Annahme, dass alle Prozesslaufzeiten und Prozessinstanzdaten in dem WfMS gespeichert werden können, hat man sich auf das Markieren der unterschiedlichen Schnittstellen konzentriert. Für die Darstellung wurden zur Markierung der Fremdsysteme verschiedene Linientypen verwendet:

..... : Host und seine Programmmodule  
 -----: Remote-Anbindung zu einem anderen System (Schufa)

Wurden die Schnittstellen analysiert und mit der Recherche nach vorhandenen Applikationen und Teillösungen abgeglichen, erfolgt die Kontrolle, ob die benötigten Funktionalitäten bereits vorhanden sind. Ist dies nicht der Fall, so müssen die technisch noch nicht unterstützten Schnittstellen sehr detailliert auf der technischen Ebene beschrieben werden. Da der Aufwand dafür leicht unterschätzt wird, empfiehlt es sich, für jede Schnittstelle eine genaue Konzeption zu erstellen, z.B. in Form von einem separaten Aktivitätsdiagramm und einem Sequenzdiagramm (vgl. Kapitel 4 Prozessmodellierung). Diese Diagramme müssen alle Attribute und Aktionen enthalten, die an die Schnittstelle übergeben und von der Schnittstelle erwartet werden. Hierzu wird zusätzlich ein Datenflussplan (vgl. Abbildung 6.10)

für jede Schnittstelle erstellt, in dem die Daten gruppiert nach den Aktionen und den Funktionen beschrieben werden. Für die spätere Realisierung muss darauf geachtet werden, dass die Aktionen in ihrer technischen Realisierung als Funktionsaufrufe oder Nachrichten so allgemein wie möglich gehalten werden. Nur so kann gewährleistet werden, dass die einzelnen Funktionen der Schnittstellen im Rahmen eines modularen Workflow-Konzeptes wiederverwendet werden können.

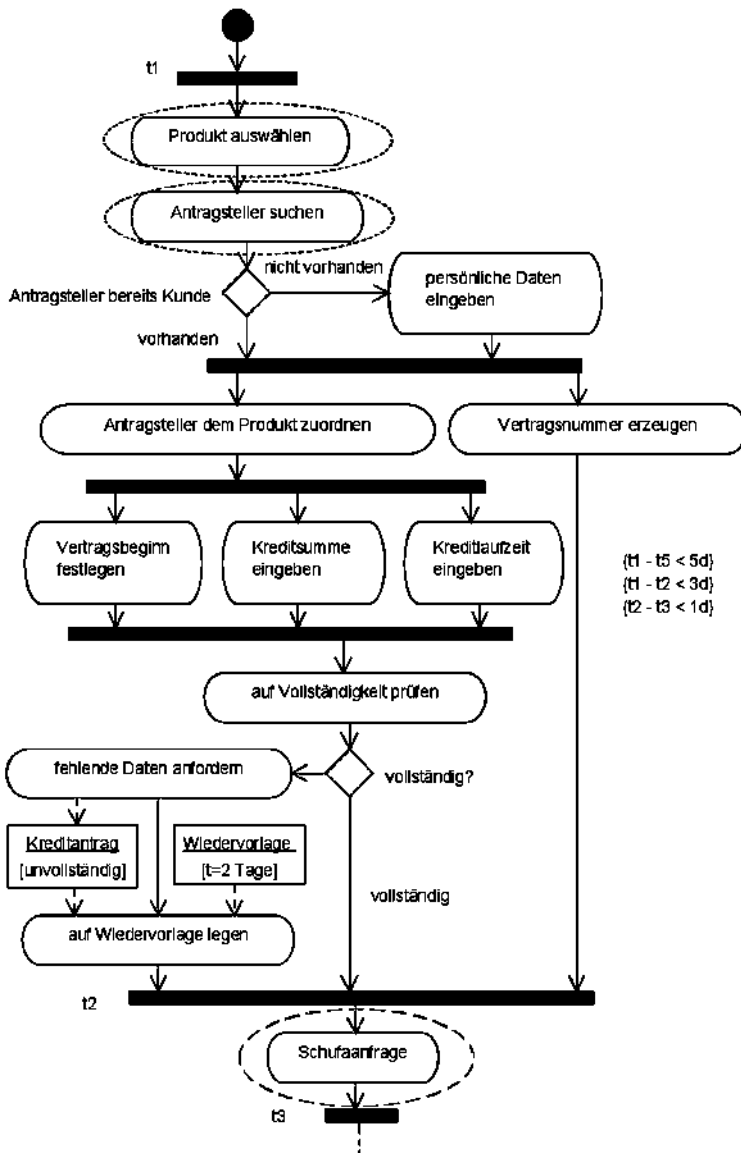
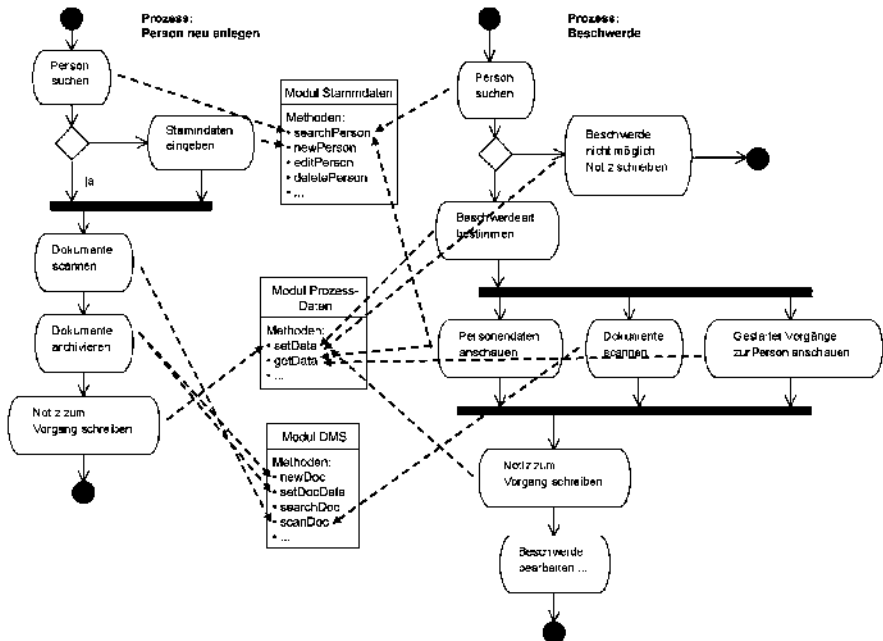


Abbildung 6.7. Aktivitätsdiagramm mit Anwendungen

Es ist wichtig, nach der Modellierung des Geschäftsprozesses den Aktivitäten die Funktionen zuzuordnen (vgl. Abbildung 6.8), um anschließend die Module zusammenstellen zu können. Die Modularisierung des Systems wird in Zusammenarbeit mit dem technischen Architekten und dem Prozessmodellierer erarbeitet. Der technische Architekt bereitet dann aus dieser und aus den sich abzeichnenden späteren Anforderungen die Architektur so auf, dass der Modellierer für die Definition der folgenden Geschäftsprozesse auf eine Auswahl (Pool) von Aktivitäten zugreifen kann. Diese sind dann für weitere Aktivitäten schon mit den jeweiligen Funktionen verknüpft.



**Abbildung 6.8.** Funktionale Aufteilung der Module

Dies hat zur Folge, dass nach einer relativ kurzen Zeit der Einführung eines WfMS nur noch geringer Aufwand für die technische Realisierung weiterer Geschäftsprozesse entsteht, da diese hauptsächlich durch die Kombination bereits existierender Aktivitäten aus dem Pool implementiert werden können. In Abbildung 6.9 wird ein solcher Pool dargestellt. Mit diesem kann der Modellierer nur mit einem Modellierungswerkzeug einen Geschäftsprozess erstellen und diesen dann mit dem vollen Funktionsumfang für die Sachbearbeiter zur Verfügung stellen, ohne eine Veränderung bzw. Erweiterung an der Funktionsschicht vornehmen zu müssen.

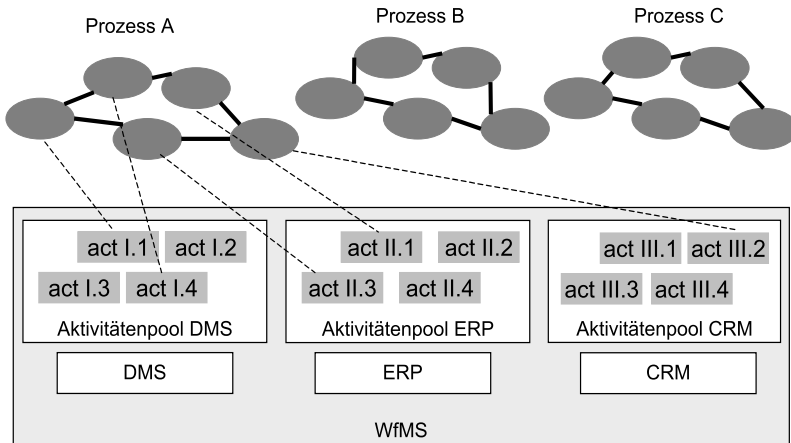


Abbildung 6.9. Aktivitätenpool

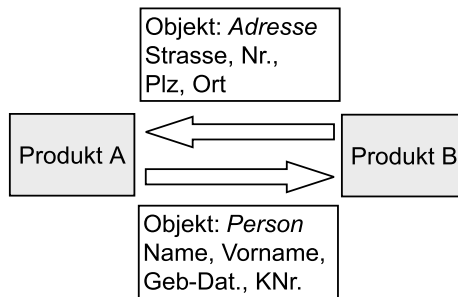


Abbildung 6.10. Datenflussplan einer Schnittstelle

### WfMS

Wie vor jeder Investition in eine Software-Lösung, so muss auch für ein WfMS-Produkt eine Evaluierung durchgeführt werden. Als Basis für eine Evaluierung werden die Auswahlkriterien für ein WfMS aufgenommen und die vorhandene IT-Infrastruktur analysiert. Es ist dabei zu beachten, dass sowohl die Hardware als auch die Software, die für die Lösung verwendet werden soll, berücksichtigt wird. Weitere Auswahlkriterien sind im Kapitel Werkzeuge/Auswahlkriterien für ein WfMS aufgeführt.

Auswahl von Fragen zur Basisaufnahme:

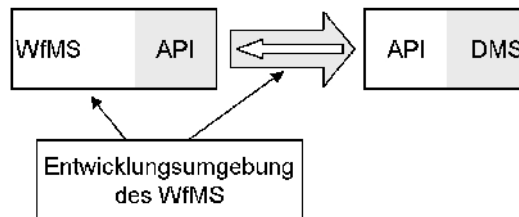
- Wurden in dem Unternehmen schon webbasierte oder Client/Server-Lösungen implementiert, auf denen man ggf. aufbauen kann?
- Ist die Kapazität der Netzwerktopologie schon an ihre Grenzen gestoßen?
- Welche strategischen Softwarekomponenten gibt es (Server-Betriebssystem, Datenbank, Client-Betriebssystem)?

- Welche Software-Lösungen und Software-Produkte befinden sich im Unternehmen im Einsatz?
  - Werden die genannten Lösungen oder Produkte für die Prozesse benötigt?
  - Haben die benötigten Lösungen oder Produkte eine Möglichkeit der Anbindung? Wenn ja, welche?
  - Welches Lizenzmodell wird für die benötigten Produkte verwendet?
  - Ist eine Ablösung oder Aktualisierung der Produkte bzw. der Lösungen geplant?
  - Ist bereits eine Middleware im Unternehmen vorhanden?
- Gibt es ein Autorisierungs- und Authentifizierungssystem?

In einem weiteren Schritt wird der zuvor modellierte Prozess mit den globalen Zielen des Unternehmens abgeglichen. Da eine große Auswahl von WfMS-Produkten mit unterschiedlichen Einsatzschwerpunkten am Markt vorhanden ist, muss man sehr genau die Ziele und Anforderungen des eigenen Unternehmens kennen, um diese mit einem WfMS-Produkt ideal abdecken zu können.

### Entwicklungsumgebung

Die Entwicklungsumgebung ist hauptsächlich abhängig von der Auswahl des WfMS. Es gibt bereits Hersteller, die eine eigene, spezialisierte Entwicklungsumgebung mit ihrem WfMS-Produkt ausliefern. Diese beschränkt sich in der Regel auf die Realisierung der Prozesse innerhalb des WfMS, d.h. der Entwickler kann die eine Seite der Schnittstellen mit der Entwicklungsumgebung des Produktes realisieren, muss aber für die andere Seite eine „konventionelle“ Entwicklungsumgebung verwenden, um beispielsweise ein DMS über dessen API anzubinden. Der in Abbildung 6.11 hell dargestellte Pfeil wird durch die mitgelieferte Entwicklungsoberfläche des WfMS-Produktes realisiert. Der dunkle Pfeil stellt die DMS-API dar. Diese API wird i.d.R. über eine separate Entwicklungsumgebung programmiert; diese kann Bestandteil des DMS sein.



**Abbildung 6.11.** Beispiel einer Anbindung eines DMS an ein WfMS

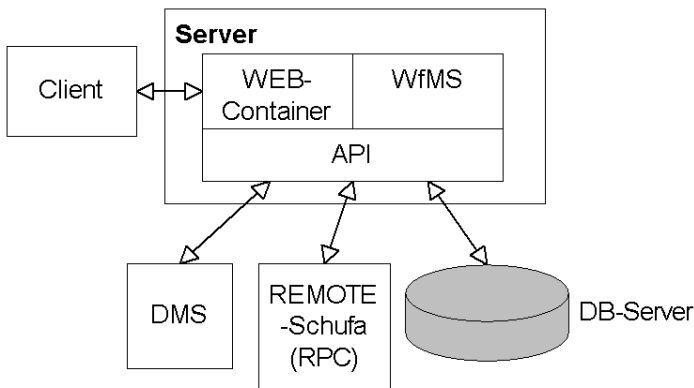
Durch die Entscheidung für ein WfMS-Produkt legt sich das Unternehmen auf die Art und Weise der Anbindung und die Entwicklungssprache fest. Es gibt eine Vielzahl von Entwicklungssprachen, mit denen ein WfMS programmiert und erweitert werden kann. Hier nur zwei Produktbeispiele, welche in dem Kapitel Werkzeuge näher betrachtet werden.

Das Produkt eWork der Firma Metastorm GmbH hat einen eigenen grafischen Designer für die Modellierung der Geschäftsprozesse. Möchte man nun ein externes System anbinden, so besteht die Möglichkeit, dies über die integrierte Visual-Basic-Schnittstelle oder über Skripte zu realisieren. Entscheidet man sich für Visual Basic, so werden Teile des Visual-Basic-Sourcecodes in dem grafischen Designer realisiert, während andere Teile in einer Entwicklungsumgebung eines Drittanbieters, wie z.B. Visual Studio von Microsoft, realisiert werden müssen.

Das WfMS-Produkt Business Manager der Firma Savvion liefert zwar auch einen eigenen grafischen Designer für die Modellierung der Prozesse, verfolgt aber aufgrund der J2EE-Produktarchitektur den Ansatz, dass weitere Applikationen über Java-Skripte oder über EJBs angebunden werden können. Das JavaScript wird im grafischen Designer realisiert und die EJBs werden in einer Entwicklungsumgebung eines Drittanbieters, wie z.B. JBuilder von Borland, implementiert.

Die oben aufgeführten Produktbeispiele verdeutlichen, dass abhängig von der WfMS-Produktauswahl die Möglichkeiten der Anbindung unterschiedlich sein werden. Eine Entwicklungsumgebung muss demnach während der Evaluierungsphase als Auswahlkriterium mit berücksichtigt werden (vgl. Kapitel 8 Werkzeuge).

Als Ergänzung zu den oben gezeigten Diagrammen und dem Datenflussplan ist noch eine Architekturübersicht für die technische Umgebung zu erstellen. Diese dokumentiert die bestehende Infrastruktur und beschreibt die benötigten Applikationen und deren Verknüpfung mit dem WfMS. In Abbildung 6.12 findet der Leser eine Übersicht aller Applikationen, welche für den Beispielprozess analysiert wurden.



**Abbildung 6.12.** Beispielarchitektur

6.2.6 Geschäftsprozess implementieren

Bevor mit der Implementierung begonnen wird, muss eine separate Entwicklungsumgebung aufgebaut werden (vgl. Abbildung 6.13). Mittelständische Unternehmen haben eine solche Umgebung i.d.R. als festen Bestandteil ihrer IT-Infrastruktur, während kleinere Unternehmen diesen Aufwand scheuen.

Da die Realisierung des Geschäftsprozesses starken Einfluss auf das Kerngeschäft nimmt, sollten die Kosten für eine Entwicklungsumgebung, auch wenn sie nur für den Zeitraum der Realisierung benötigt wird, aufgebracht werden. Wird keine Entwicklungsumgebung vorgesehen, können durch auftretende Fehler in dem neu erstellten Prozess Teile des Kerngeschäftes ausfallen. Der so entstandene Schaden ist in der Regel höher als der Aufwand für eine separate Entwicklungsumgebung.

Mit dieser Phase beginnt die Realisierung des Geschäftsprozesses, und somit muss entschieden werden, mit welchem Freiheitsgrad der Geschäftsprozess implementiert werden soll. Bisher wurde immer der Idealfall betrachtet und dokumentiert, d.h. es wurde das Optimum an Automatisierung und Funktionalität beschrieben. Aus den unterschiedlichsten Gründen kann es aber von Interesse sein, dass die Implementierung in mehreren Schritten erfolgt. Die wichtigsten Gründe hierfür sind ein früher Return On Investment (ROI) und die Möglichkeit, durch eine frühe Inbetriebnahme Erfahrungen mit dem eingeführten Geschäftsprozess zu erhalten. Diese sind dann in späteren Phasen von Nutzen. In Tabelle 6.2 ist eine Auswahl der bekanntesten Ausprägungen aufgeführt.

Tabelle 6.2. Ausprägungen der Implementierung

Ausprägung	hoher Aufwand	geringer Aufwand
Funktionalität und Automatisierung	WfMS übernimmt alle Funktionalitäten und führt diese soweit wie möglich automatisch aus	Es werden nur die Dokumente an den Geschäftsprozess angehängt.
Speicherung der Daten	zentral	dezentral (lokal)
Detaillierungsgrad des implementierten Geschäftsprozesses	Es werden alle möglichen Varianten implementiert	Es wird nur der Regelfall implementiert.

Während der Realisierung kann auch mit der Schulung der Sachbearbeiter und der Administratoren begonnen werden, da zu diesem Zeitpunkt das WfMS und der Geschäftsprozess feststehen. Die geschulten Sachbearbeiter können so die folgende praxisorientierte Testphase effektiv unterstützen und zu einer schnellen Fehlerlokalisierung und Fehlerbeseitigung beitragen. Da die Schulung in diesem Fall parallel zu der Realisierung durchgeführt wird, hat dies auf die Dauer des Projektes keinen negativen Einfluss.

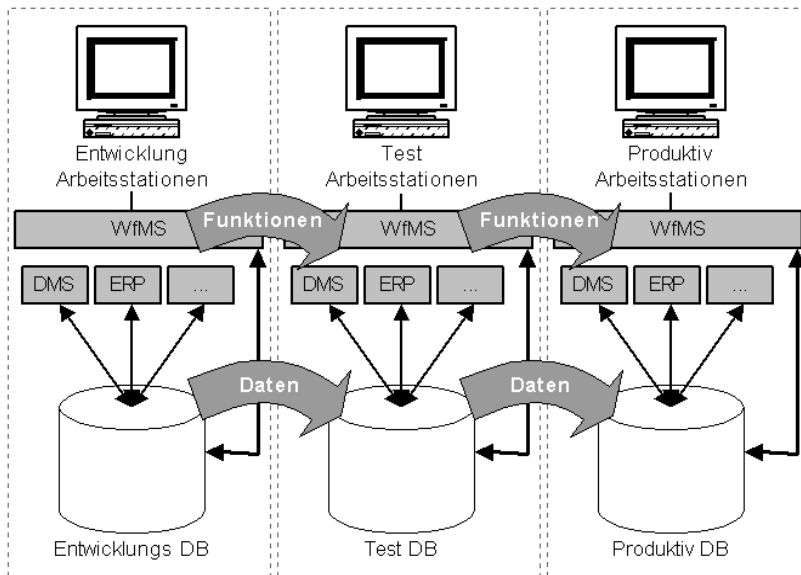


Abbildung 6.13. Unterschiedliche IT-Umgebungen

### 6.2.7 Test des Geschäftsprozesses

Für den Test des Geschäftsprozesses bzw. der Lösung werden die zuvor erstellten Aktivitätsdiagramme als Grundlage für das Ausarbeiten von Testszenarien verwendet. Es ist wichtig, dass neben den fachlichen Testszenarien auch die technischen Tests durchgeführt werden. Hierzu gehören die Funktions-, Performance- und die Integrationstests. Da die Anforderungen an den Geschäftsprozess schon zu einem frühen Zeitpunkt des Projektes feststehen, kann frühzeitig mit der Testvorbereitung und dem Ausarbeiten der Testszenarien begonnen werden.

Die Personen bzw. Rollen, welche die fachlichen Anforderungen während der Modellierung definiert haben, sind auch für die Ausarbeitung der Testszenarien und letztendlich auch für die Durchführung der Tests vorzusehen. Für den fachlichen Test empfiehlt es sich, eine Testumgebung zur Verfügung zu stellen, welche der Produktivumgebung entspricht und einen realistischen Testbetrieb zulässt. Im Gegensatz zu den fachlichen Tests können die technischen Tests, also die Tests der Schnittstellen zu Fremdsystemen, vorab durch die Entwicklung mit kleinen Testprogrammen durchgeführt werden. Da diese unabhängig von dem Gesamttest sind, werden die Entwickler bzw. die Zulieferer in die Pflicht genommen, diese Tests durchzuführen. Der eigentliche Gesamttest wird in der Testumgebung durchgeführt. Im Idealfall besteht dabei keine topologische Verbindung zwischen den IT-Netzen (Entwicklungs-, Test- und Produktivnetz). Der Mehraufwand für eine Trennung wird als sehr hoch beziffert. Wird aber keine einwandfreie Trennung der Netze durchgeführt und u.U. sogar in einem Produktivnetz entwickelt, so birgt das die Gefahr, den Ablauf der aktuellen Kerngeschäftsprozesse insbesondere bei der



Weiterentwicklung zu behindern. Würde ein solcher Fall eintreten und man dann die Ausfallkosten dieser Geschäftsprozesse dagegen rechnen, so würden sich die Kosten für eine Trennung der Netzwerke bzw. der Aufbau einer Entwicklungs-, Test- und Produktivumgebung relativieren. In Abbildung 6.13 ist eine konsequente Trennung der IT-Umgebungen dargestellt.

### **6.2.8 Geschäftsprozess freigeben**

Vor der Freigabe des Geschäftsprozesses ist dafür zu sorgen, dass alle technischen Voraussetzungen in der Produktionsumgebung geschaffen wurden. Um den sicheren Betrieb gewährleisten zu können, ist für die Abteilung, die für den Betrieb verantwortlich ist, ein Betriebshandbuch zu erstellen und für den Systembetrieb zu übergeben. In diesem Betriebshandbuch werden die Installation, die technische Überwachung (Monitoring) und die Tätigkeiten der einzelnen Anwender sowohl während des normalen Betriebes als auch im Fehlerfall beschrieben. Eine grobe Inhaltsbeschreibung eines Betriebshandbuches ist im Kapitel Template aufgeführt.

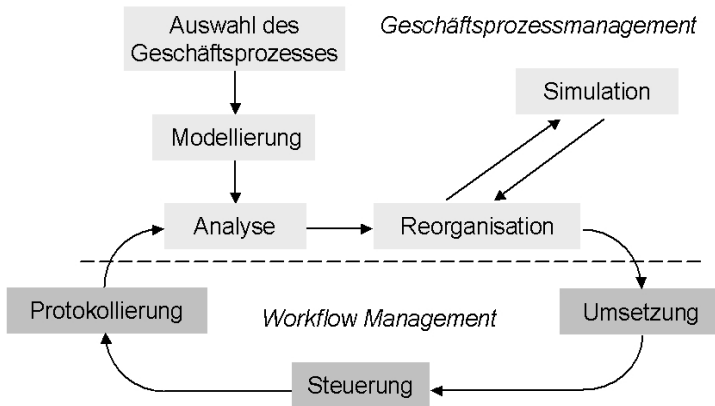
Die Freigabe des Geschäftsprozesses erfolgt auf der fachlichen Seite durch die Fachabteilung und auf der technischen Seite durch die Administratoren. Eine Freigabe darf erst dann erfolgen, wenn alle Tests positiv verlaufen sind und somit gewährleistet ist, dass durch die Inbetriebnahme des Geschäftsprozesses kein Schaden für das Unternehmen entsteht. Für das Freigabeverfahren empfiehlt sich die Erstellung von Freigabe- bzw. Übergabeprotokollen (vgl. T-Wfbl-013), in denen die Ergebnisse einschließlich eventueller Mängel beschrieben werden.

Wird der Geschäftsprozess freigegeben, kann mit der Installation in der Produktionsumgebung begonnen werden. Es empfiehlt sich solche Installationen in betriebsarmen Zeiten über Nacht oder an Wochenenden durchzuführen, um den laufenden Betrieb nicht zu behindern. Für den Fall, dass die Inbetriebnahme aufgrund von erheblichen Mängeln des Geschäftsprozesses rückgängig gemacht werden muss, sind entsprechende Fallback-Strategien bereitzustellen. Zwar ist das Risiko hierfür gering, tritt es dennoch ein, können die Auswirkungen und der daraus resultierende Schaden bei Nichtberücksichtigung sehr hoch sein.

### **6.2.9 Laufzeitdaten erfassen**

Nach der Einführung eines WfMS ist für die kontinuierliche Optimierung der Prozesse mit Hilfe des Workflow-Management-Regelkreises (vgl. Abbildung 6.14) die Aufzeichnung von Laufzeitdaten von großer Bedeutung. Diese Daten bilden die Basis für die Reorganisation und Optimierung der produktiven Prozesse. Ist man vor der Einführung eines WfMS noch von relativ ungenauen Zahlen ausgegangen, die durch eine aufwändige Erfassung und/oder durch Schätzen der Zeiten entstanden sind, so kann man durch die Aufzeichnung der tatsächlichen Prozessdaten mit genauer Zeiterfassung die Optimierungen durchführen. Es ist bei der Evaluierung des WfMS darauf zu achten, dass die WF-Engine diese Möglichkeiten der Datenerfassung bietet, die Zeiten standardisiert aufzeichnet und die Daten für eine Auswertung zur Verfügung stellt. Die Laufzeitdaten werden über einen

sinnvollen Zeitraum gesammelt, um diese später dem Modellierungs- bzw. Simulationswerkzeug zuzuführen. Der auszuwertende Erfassungszeitraum ist dabei so zu wählen, dass ausreichend Geschäftsprozesse beendet wurden und diese auch das durchschnittliche Arbeitsaufkommen repräsentieren. Dieser Zeitraum bewegt sich in einem Bereich zwischen vier Wochen und sechs Monaten.



**Abbildung 6.14.** Workflow-Management-Regelkreis

### 6.2.10 Auswerten der Laufzeitdaten und Redesign des Prozesses

Im Idealfall werden die Laufzeitdaten für die Auswertung an ein Simulationsprogramm übergeben. Mit diesem können die aufgezeichneten Daten im Prozessdiagramm dargestellt und der Prozess auf Schwachstellen analysiert werden. Diese Schwachstellen zeichnen sich z.B. durch eine hohe Liegezeit vor und nach der Bearbeitung aus, oder durch eine sehr hohe Bearbeitungszeit. Auch Bearbeitungsengpässe können erkannt und organisatorisch oder durch Redesign des Prozesses behoben werden.

Nach der Analyse ist zu überlegen, welche Maßnahmen getroffen werden können, um jene Aktivitäten zu optimieren, die die meiste Zeit innerhalb des Geschäftsprozesses benötigen.

Mögliche Ursachen ineffizienter Prozesse sind:

- Die Anzahl der fehlerhaften Geschäftsprozesse ist hoch. Dies kann mehrere Gründe haben. Zum einen können die Mitarbeiter nicht ausreichend geschult sein. Zum anderen besteht die Möglichkeit, dass der Geschäftsprozess nicht komplett von dem WfMS unterstützt wird und es zu Medienbrüchen kommt. Diese haben zur Folge, dass die Mitarbeiter eine zu hohe Handlungsfreiheit besitzen, welche die Fehleranfälligkeit während der Bearbeitung erhöht.
- Keine klare Definition der Rollen, oder ein Mitarbeiter hat mehrere Rollen. Dies tritt häufig in kleinen Unternehmen auf, in denen ein Mitarbeiter, z.B.

aufgrund von Personalmangel, mehrere Rollen inne hat, aber auch die „Übernahme“ von Aufgaben auf dem kleinen Dienstweg durchgeführt wird.

- Schlechte bzw. unnötige Arbeitsverteilung auf mehrere Mitarbeiter; dadurch entstehen hohe Liegezeiten.

Sind die Schwachstellen identifiziert, dann wird eine Entscheidung getroffen, ob eine Veränderung im organisatorischen Ablauf oder in der technischen Unterstützung des Geschäftsprozesses die Mängel beseitigen soll. Um eine Entscheidung herbeizuführen bzw. zu untermauern, kann der Geschäftsprozess in dem Modellierungswerkzeug verändert und mit dem Simulationswerkzeug mit einem Teil der Laufzeitdaten simuliert werden. Ergibt die Simulation, dass der Geschäftsprozess weiter überarbeitet werden muss, so empfiehlt es sich, Teile des hier beschriebenen Vorgehensmodells zu wiederholen. Die Durchführung der einzelnen Phasen muss nicht mit dem gleichen Aufwand betrieben werden wie in der ersten Iteration, doch sollte jede Phase durchlaufen werden, um die Erfolgswahrscheinlichkeit zu erhöhen.

### **6.3 Projektorganisation und Projektrollen**

Die Projektorganisation in Workflow-Projekten unterscheidet sich nicht wesentlich von der Organisation anderer IT-Projekte. Es gilt nur auf ein paar spezielle Eigenheiten zu achten, welche im Folgenden beschrieben werden.

Bevor weitere Angaben zu der Organisation des Projektes gemacht werden können, muss zuerst der Ausgangspunkt bestimmt werden. Der Umfang der angestrebten Workflow-Einführung ist hier ein entscheidender Faktor für die Auswahl der beteiligten Rollen. Wird z.B. eine unternehmensweite Workflow-Einführung in mehreren Phasen geplant, so muss die Geschäftsführung die Entscheidung für ein Workflow-Produkt erheblich mittragen. Wird aber nur geplant, einen kleinen abteilungsinternen Geschäftsprozess zu realisieren, reicht es je nach Unternehmen meist aus, die Geschäftsführung über die Auswahl des Workflow-Produktes zu informieren. Neben dem Umfang der Workflow-Einführung sind die im Unternehmen vorhandenen Ressourcen für den Einsatz der benötigten Rollen entscheidend. Es sind rechtzeitig die Ressourcen zu buchen, um eine Verzögerung durch verspätete Einsetzbarkeit zu vermeiden.

Der folgende Abschnitt orientiert sich an dem zuvor beschriebenen Vorgehensmodell. Hier werden die Rollen den entsprechenden Fähigkeiten oder Anforderungen zugeordnet. Begonnen wird mit der Initialisierungsphase:

Phase: Initial

Beteiligte Rollen: Geschäftsführung, Abteilungsleiter der primär betroffenen Fachabteilung

Beschreibung:

Ist es geplant, unternehmensweit die Prozesse mit einer Workflow-Engine zu unterstützen, dann hat die Geschäftsführung diesen Auftrag zu erteilen. Dies nicht

nur, da das Projekt eine unternehmensweite Bedeutung besitzt, sondern auch, weil die finanziellen Mittel nicht auf eine Kostenstelle gebucht werden können. Ist keine unternehmensweite Workflow-Einführung geplant, so muss die Geschäftsführung trotzdem eingeschaltet sein, da sie auf die Einhaltung der Unternehmensziele zu achten hat.

Neben den Unternehmenszielen müssen auch die „kleineren“ Ziele verfolgt werden. Diese werden in kleinen Schritten geplant und führen in kurzen Zeitabschnitten zu Ergebnissen. In dieser Verantwortung steht der Abteilungsleiter, der durch die Nähe zum Tagesgeschäft die aktuelle Ist-Situation sehr gut einschätzen kann.

Phase: Analyse der bestehenden Prozesse

Beteiligte Rollen: Sachbearbeiter, Abteilungsleiter der Fachabteilungen  
Beschreibung:

Es werden die Sachbearbeiter sein, die im Tagesgeschäft mit der Bearbeitung der Geschäftsprozesse eingebunden sind. Die Geschäftsprozesse sind im Allgemeinen vom Ablauf her nur organisatorisch definiert.

Die Abteilungsleiter haben in dieser Phase nur eine verbindende und kontrollierende Funktion zwischen den Abteilungen.

Phase: Projektziel bestimmen

Beteiligte Rollen: Geschäftsführung und Abteilungsleiter der Fachabteilung

Beschreibung:

Die beteiligten Rollen müssen sich einig über das zu erreichende Ziel sein. Die Ziele des Abteilungsleiters müssen mit denen der Geschäftsführung kooperieren bzw. es muss eine Annäherung auf der Basis der bestehenden Geschäftsprozesse erreicht werden.

Phase: Modellieren der Prozesse

Beteiligte Rollen: Sachbearbeiter, Modellierer, Prozessverantwortlicher

Beschreibung:

Die ausgewählten Prozesse werden von den drei Rollen bearbeitet. Der Modellierer muss über das entsprechende Know-how des betroffenen Prozesses verfügen und die möglichen Tools kennen/beherrschen, welche die Anforderungen abdecken können. Der Sachbearbeiter ist dafür verantwortlich, dass die fachlichen Anforderungen erfüllt werden. In diesem frühen Stadium des Projektes ist es schon wichtig, einen Prozessverantwortlichen zu benennen. Dieser wird während der „Lebenszeit“ des Projektes (des Prozesses) der zentrale Ansprechpartner für fachliche Entscheidungen sein.

Phase: Technische Konzeption und Feinmodellierung

Beteiligte Rollen: Sachbearbeiter, Modellierer, Architekt

Beschreibung:

Im Mittelpunkt dieses Schrittes steht der Architekt, der für die technische Realisierungskonzeption verantwortlich ist.

Phase: Implementierung

Beteiligte Rollen: Architekt, Entwickler für das Workflow-Produkt und die

Workflow-Anbindung, Entwickler für fachliche Lösungen(z.B. SAP-Entwickler)  
Beschreibung:

Während der Implementierung hat der Architekt eine kontrollierende Rolle. Er ist dafür verantwortlich, dass die zuvor entworfene Architektur realisiert wird und steht somit in engem Kontakt mit den Entwicklern. Die Entwickler unterscheiden sich in ihren Fähigkeiten zwischen WfMS-produkterfahren, erfahren in der Schnittstellenrealisierung und erfahren in der Realisierung fachlicher Anwendungen.

Phase: Test

Beteiligte Rollen: Alle Arten von Entwicklern, Sachbearbeiter

Beschreibung:

Der Sachbearbeiter hat während der Tests die Verantwortung für die korrekte Ausführung der realisierten Prozesse. Kommt es zu einem nicht definierten Zustand bzw. Ablauf der Prozesse, korrigieren sie zusammen mit den Entwicklern diese Fehler. Die Unterstützung des Sachbearbeiters beschränkt sich auf die Entdeckung und Beschreibung des Fehlers und der korrekten Funktionsweise.

Phase: Freigeben

Beteiligte Rollen: Sachbearbeiter, Fachabteilungsleiter, Prozessadministrator, IT-Administrator

Beschreibung:

Der Sachbearbeiter beantragt zusammen mit dem Fachabteilungsleiter die Freigabe für die neu erstellten oder überarbeiteten Prozesse. Da die beiden Administratoren für den Betrieb der Prozesse verantwortlich sind, müssen diese in den Freigabeprozess mit eingebunden werden. Die Rollen sind aber unterschiedlicher Art. Der Prozessadministrator ist für den fachlich korrekten Ablauf verantwortlich, d.h. er kontrolliert die Logfiles des WfMS und eventuell die der Schnittstellen zu den anderen Systemen. Der IT-Administrator ist für den einwandfreien Betrieb der Plattform verantwortlich, auf der die Lösung installiert wurde. In Unternehmen kann es vorkommen, dass beide Rollen (Prozessadministrator und IT-Administrator) von einer Person abgedeckt werden können. Es ist auf jeden Fall erforderlich, dass im Rahmen des Projektes ein Betriebshandbuch erstellt wird, in dem der fachliche und der IT-technische Teil beschrieben ist.

Phase: Erfassen der Laufzeitdaten

Beteiligte Rollen: IT-Administrator

Beschreibung:

Phase: Auswerten der Laufzeitdaten

Beschreibung:

Die aufgezeichneten Laufzeitdaten werden in regelmäßigen Abständen durch den Prozessverantwortlichen kontrolliert. Der Rhythmus der Kontrollen ist individuell an den Prozess angepasst: wird ein Prozess täglich ca. 100mal initiiert, so ist es sinnvoll, in monatlichen Abständen die Daten zu kontrollieren. Handelt es sich aber um einen Prozess, der wöchentlich einmal gestartet wird, erhält man bei einer monatliche Auswertung keine aussagekräftigen Daten und muss einen größeren Zeitabstand wählen. Die in dieser Phase beteiligten Rollen entscheiden dann ge-

meinsam, ob ein Redesign des Prozesses sinnvoll ist oder ob andere Maßnahmen zur Optimierung ergriffen werden können.

Phase: Redesign

Beteiligte Rollen: Sachbearbeiter, Modellierer

Beschreibung:

Der Sachbearbeiter und der Modellierer überarbeiten aufgrund der zuvor aufgetragenen Prozesslaufzeitdaten den Prozess. Der Modellierer testet den überarbeiteten Prozess in einem Simulationstool und kontrolliert, ob die angestrebten Verbesserungen erreicht werden. Kommt es zur Entscheidung, den überarbeiteten Prozess zu realisieren, beginnt der Projektablauf wieder in der Phase „Technische Konzeption und Feinmodellierung“.

## 7 Ausblick

Neben der Kenntnis der derzeit auf dem Markt erhältlichen WfMS-Produkte sowie der aktuellen Technologien, mit denen die IT-Lösungen realisiert werden können, ist es auch wichtig zu erkennen, welche Technologien zukünftig in den Produkten verwendet werden können. Außerdem soll betrachtet werden, welche Alternativen zu den konventionellen Produkten existieren.

### 7.1 Web Services

Die Anwendungsentwicklung wird aufgrund der stetig wachsenden Anforderungen immer komplexer und verlangt ein umfangreiches Wissen über Technologien und Vorgehensweisen. Zur Verringerung des Realisierungsaufwandes der Anwendungen bietet sich die Einbindung von Web Services an. Diese ermöglichen es, bereits realisierte Funktionalitäten in Form von Diensten (Services) über das Internet in die Eigenentwicklung zu integrieren. Um dies auch bei unternehmenskritischen Anwendungen zu ermöglichen, müssen Aspekte wie Stabilität, Sicherheit und Standardisierung aller beteiligten Ressourcen gewährleistet werden.

Web Services müssen nicht zwangsläufig durch einen externen Anbieter bereitgestellt werden. Es besteht genauso die Möglichkeit, innerhalb eines Unternehmens fachliche Module als Web Services anzubieten und diese in weitere Anwendungen zu integrieren. Ein großer Vorteil von Web Services ist, dass ein solcher Service von einem anderen Web Service aufgerufen werden kann. Dadurch können die Methoden des aufgerufenen Web Service so genutzt werden, als wären sie ein Bestandteil in der internen IT-Architektur eingebundener Module. Dies wird durch die in einem Standard definierte Schnittstelle der Web Services ermöglicht: ein Dritter kann den Web Service aufrufen, ohne die dahinter liegende Software-Architektur zu kennen. Somit können Web Services im Zusammenhang mit verteilten Anwendungen, Middleware und Integrations-Servern gesehen werden.

Da Web Services nach außen in erster Linie aus Spezifikationen bestehen, ist es wichtig, dass ein unabhängiges Konsortium für die Einhaltung dieses Standards und die Verabschiedung weiterer Standards verantwortlich ist. Im Fall von Web Services übernimmt das World Wide Web Consortium (W3C) diese Aufgabe.

### 7.1.1 Definierte Rollen

Die Definition von Web Services sieht drei verschiedene Rollen vor. Für die Realisierung einer Verbindung werden mindestens zwei beteiligte Parteien benötigt, auf die diese drei Rollen verteilt werden. Das bedeutet, dass die Rolle eines Service Providers und eines Service Brokers durch ein und dasselbe Unternehmen erfolgen kann.

#### Dienstanbieter (Service Provider)

Der Service Provider implementiert, betreibt und pflegt den Web Service, der über das Internet angeboten wird. Für das Auffinden im Internet wird der Service durch Dienstanbieter beschrieben.

#### Dienstinachfrager (Service Requestor)

Der Service Requestor fordert einen Service an, der in seine Anwendung oder auch in seine eigenen Dienste integriert wird. Zu diesem Zweck sucht er einen Service, der die von ihm spezifizierten Anforderungen erfüllt.

#### Dienstmakler (Service Broker)

Der Service Broker vermittelt die Web Services. Zu diesem Zweck werden die Beschreibungen der Web Services gespeichert und nach Kategorien indexiert auf eigenen Servern verwaltet. Der Dienstmakler ermöglicht außerdem das automatisierte Auffinden existierender Web Services, die nicht nur in seiner Datenbank registriert sind, sondern noch von weiteren Service Brokern angeboten werden.

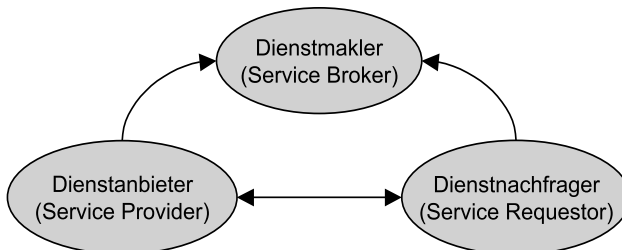


Abbildung 7.1. Web-Service-Rollen

### 7.1.2 Technik

Web Services kommunizieren über HTTP und können sowohl über das interne Firmennetzwerk als auch über das World Wide Web angesprochen werden. Sie sind leicht in die bestehenden Infrastrukturen integrierbar, da zur Kommunikation verbreitete Standards wie das bereits erwähnte HTTP und XML (für den Austausch von Informationen) eingesetzt werden.

Für ein besseres Verständnis von Web Services ist es wichtig zu erkennen, dass es sich um eine Kombination aus Protokollen und Spezifikationen handelt. Diese werden Im Folgenden genannt.



**HTTP (Hypertext Transfer Protocol)**

Bei HTTP handelt es sich um das zwischen Web-Browser und Webserver eingesetzte Datentransferverfahren für Anfragen des Clients sowie Antworten des Servers.

**XML (Extensible Markup Language)**

XML ist eine Spezifikation zur Modellierung von Datenstrukturen mit semantischen Anteilen (Feldbezeichner) sowie Validierungs- und Auswertungsmechanismen (Dokumententyp-Definitionen, Style-Sheets, Pfadspezifikation usw.). Der Vorteil dieses Beschreibungsformats liegt darin, dass es einerseits von Menschen gut gelesen und andererseits von Maschinen gut verarbeitet werden kann.

**SOAP (Simple Object Access Protocol)**

SOAP ist ein standardisiertes Kommunikationsprotokoll auf Basis von XML für das Versenden und Empfangen von Nachrichten über das Web. Die Kommunikation zwischen dem jeweiligen Service Provider und Service Requestor erfolgt über SOAP-Messages. Diese Messages enthalten die Funktionsaufrufe und die zu übermittelnden Daten.

Da SOAP leicht zu implementieren ist, existieren bereits heute mehrere Implementierungen für Apache, WebSphere, Java und Visual Basic. Das Protokoll wurde als Standard durch die W3C eingestuft.

**WSDL (Web Services Description Language)**

Durch die WSDL wird ein Web Services genau beschrieben: das verwendete Protokoll, die Adresse und die Port Nummer, die möglichen Prozeduren und Funktionen sowie die Formate für Input und Output. Die Sprache basiert auf XML und legt die erforderlichen Datenformate fest. Die Beschreibung des Service wird bei einem Service Broker hinterlegt.

**UDDI (Universal Description, Discovery and Integration)**

UDDI beschreibt die Veröffentlichung von und die Suche nach Web Services, d.h. die Lokalisation unterschiedlicher Web Services. Die Spezifikation legt fest, wie der Web Service anhand von Eigenschaften eindeutig beschrieben wird. Anhand dieser Beschreibung kann der Web Service im Netz gefunden werden. Dafür wurde ein Verzeichnis definiert, in dem der Service Provider seinen Dienst registrieren und ein Service Requestor diesen suchen kann. Die Einordnung der Web Services erfolgt in der UDDI Registry nach Kategorien von Branchen und Unternehmen. Der UDDI ist selber als Web Service realisiert und kann aufgrund eines regelmäßigen Abgleichs zwischen allen UDDI-Standorten immer den aktuellen Stand der Web Services publizieren. Es können auch manuelle (durch den Anwender) oder automatische (durch die Anwendung) Abfragen z.B. nach dem preisgünstigsten Dienst durchgeführt werden.

**7.1.3 Web Services in Verbindung mit Workflow**

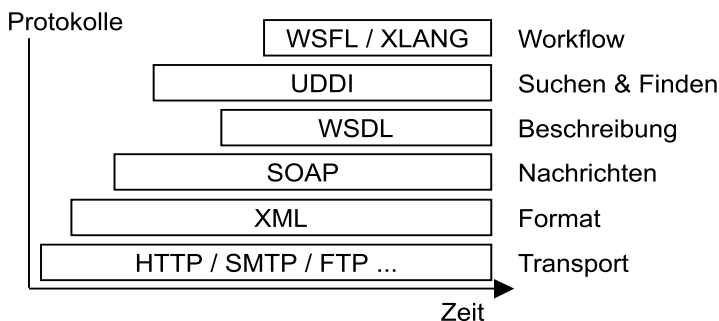
Web Services bieten aufgrund ihrer Definitionen gute Voraussetzungen für die Realisierung von Anwendungen in Verbindung mit einer Prozesssteuerung. Denk-

bar sind unterschiedliche Varianten. Zum einen bietet es sich an, eine Workflow-Engine aus Web Services zu realisieren, die die übergreifende Prozesssteuerung von mehreren Unternehmen übernehmen kann. Dies hätte aber den Nachteil, dass die modellierten Prozesse der Unternehmen an den Service Provider übergeben werden müssen und der Provider diese verwalten müsste. Der Provider hätte so einen Einblick in zum Teil unternehmenskritische Prozesse. In einer anderen Variante würden Teile der fachlichen Anforderungen als Web Service eingebunden und die Prozesssteuerung im Unternehmen implementiert werden. Diese Teile könnten Berechnungen oder Auskünfte sein, die in einem Geschäftsprozess benötigt werden. Denkbar wäre eine Schufa-Abfrage oder die Auskunft über Börsenkurse. Außerdem könnten im Falle von unternehmensübergreifenden Prozessen Module als Web Services zur Verfügung gestellt werden. Dies ist beispielsweise durch ein Versandhaus (Service Requestor) mit einem Paket-Tracking-System Service Provider bereits realisiert worden. Das Versandhaus kann in diesem Fall dem Kunden anbieten, via Internet Auskünfte über den aktuellen Versandstatus seiner Bestellung einzuholen.

Derzeit ist allerdings kein Standard des W3C-Konsortiums für ein Workflow-bezogenes Protokoll definiert. Dies schränkt die weitere zukunftsichere Entwicklung derzeit stark ein. Dem W3C liegen einige konkurrierende Vorschläge vor, die von unterschiedlichen Unternehmen entwickelt worden sind:

- WSFL (Web Services Flow Language) von IBM
- XLANG (Web Services for Business Process Design) von Microsoft
- WSCI (Web Services Choreography Interface) von Sun, Intalio, SAP und BEA
- BPEL4WS (Business Process Execution Language for Web Services) von IBM und Microsoft

Einen ersten Schritt in Richtung Standard stellt die BPEL4WS dar, die aus der WSFL und dem XLANG hervorgegangen ist. Mit ihr wollen IBM und Microsoft einen Standard für Web Services inkl. Workflow ins Leben rufen, der sich gemäß Abbildung 7.2 in den allgemeinen Web-Service-Standard einordnet.



**Abbildung 7.2.** Einordnung nach IBM und Microsoft

**WSFL (Web Services Flow Language)**

Der von IBM vorgeschlagene Standard dient zur Beschreibung von Geschäftsprozessen auf der Basis von Web Services. Ähnlich wie in den UML-Aktivitätsdiagrammen basiert er auf dem Modell eines gerichteten Graphen, in dem die Knoten die Aktivitäten darstellen, die von Web Services realisiert werden. In einem solchen Graphen werden Kontrollfluss (Volllinie) und Datenfluss (gestrichelte Linie) dargestellt. Für die Workflow-Engine werden die Graphen mit XML beschrieben, um die beschriebenen Aktivitäten ausführen zu können. Dies soll für jede Workflow-Engine möglich sein, die den WSFL-Standard „versteht“, da die Spezifikation nur die Protokolle und Standards beschreibt. Jeder definierte Workflow wird als Web Service erstellt, bindet weitere Web Services ein und ist wie jeder Web Service in der UDDI-Registry registriert. Mit der WSFL wird jede Aktivität in einem XML-Dokument beschrieben und als Web Service realisiert, der durch einen Service Provider angeboten wird. Somit definiert die WSFL zum einen das „Aussehen“ der Aktivitäten und zum anderen die Kommunikation zwischen den Aktivitäten (Web Services). IBM hat WSFL in der Plattform WebSphere bereits implementiert.

**XLANG (Web Services for Business Process Design)**

Mit dem XLANG wurde die WSDL um ein Modul erweitert, welches das Verhalten eines Web Service als Teil eines Workflows beschreibt. Microsoft hat mit ein

**WSCI (Web Services Choreography Interface)**

WSCI ist eine XML-basierte Beschreibungssprache, die das Verhalten der Web Services untereinander in Bezug auf zeitliche und ablauflogische Aspekte beschreibt.

WSCI wird in Verbindung mit Beschreibungsstandards wie WSDL eingesetzt und liefert eine übergreifende, message-orientierte Sicht auf die Interaktion von Web Services. Es beschreibt jedoch nicht das interne Verhalten eines Web Service.

**BPEL4WS (Business Process Execution Language for Web Services)**

BPEL4WS wurde von BEA, Microsoft und IBM aus den Standards XLANG und WSFL erstellt. Sie definiert die Erstellung und die Verknüpfung von Workflows mit Web Services. Unternehmen können ihre Workflows damit beschreiben, die dann über mehrere Web Services betrieben werden können. In der aktuellen Definition setzt BPEL4WS auf mehreren XML-Spezifikationen auf. WSDL-Nachrichten und XML-Schema-Typdefinitionen liefern das Datenmodell, welches für Prozesse verwendet wird, und XPath beschreibt die Datenmanipulation.

Geplant ist eine Unterstützung von BPEL4WS in dem „Jupiter“ genannten E-Business-Server von Microsoft.

**7.1.4 Kommentar**

Web Services werden in der zukünftigen Anwendungsrealisierung eine zunehmende Rolle spielen, da abzusehen ist, dass die heutigen Nachteile aufgrund der fortschreitenden Entwicklung der Technologie relativiert bzw. eliminiert werden.

Die Nachteile von Web Service gegenüber konventionell erstellten Anwendungen liegen darin, dass die Performance schlechter ist und aufgrund der zum Teil noch fehlenden Standards die Kommunikation in Einzelfällen noch angepasst werden muss. Auch die Sicherheit der transportierten Informationen ist noch nicht vollständig gewährleistet, da noch kein Verfahren entwickelt wurde, das allen Angriffen von außen standhält. Es ist aber denkbar, dass mehrere Partner in einem Zusammenschluss über eine gesicherte Verbindung ihre Anwendungen über Web Services realisieren, um so ihre unternehmensübergreifenden Prozesse bezüglich der Realisierung und der Wartung effizienter betreiben zu können.

Somit kann man sagen, dass sich der Einsatz von Web Services für vereinzelte fachliche Anwendungen zwar anbietet, aber z.Zt. noch nicht als Schlüsseltechnologie zu sehen ist.

Weitere Informationen sind auf den folgenden Seiten zu finden:

Sun Microsystems	<a href="http://java.sun.com/j2ee">http://java.sun.com/j2ee</a>
IBM	<a href="http://www-106.ibm.com/developerworks/webservices/">http://www-106.ibm.com/developerworks/webservices/</a>
Microsoft	<a href="http://msdn.microsoft.com/webservices/">http://msdn.microsoft.com/webservices/</a>
Apache	<a href="http://xml.apache.org/axis/">http://xml.apache.org/axis/</a>
WebServices Architect	<a href="http://www.webservicesarchitect.com/">http://www.webservicesarchitect.com/</a>

## 7.2 Open-Source

### 7.2.1 Was ist Open-Source?

Als erstes soll der allgemeine Irrglaube ausgeräumt werden, dass Open-Source-Anwendungen kostenlos sind. Da in der englischen Sprache „free“ sowohl für Kostenfreiheit als auch für Freiheit im eigentlichen Sinne steht, ist der Irrtum entstanden, dass „kostenlos“ die einzige positive Eigenschaft von freier Software ist. Zur Beseitigung des Irrtums hat man „free“ durch „open“ ersetzt, damit nicht „kostenlos“, sondern „offen“ damit assoziiert wird. Der Begriff „free“ wurde mit der Bedeutung im Sinne der Freiheit des Geistes und der Wissenschaft gesehen. Die Initiative wurde ins Leben gerufen, um den Einsatz der Software für jeden Zweck zu ermöglichen und den freien Zugang zum Quellcode zu erhalten. Außerdem soll jeder das Recht haben, Kopien und Weiterentwicklungen an Dritte zu verteilen.

Aus „free source“ wurde 1998 durch Todd Anderson, Chris Peterson, John „Mad-dog“ Hall, Larry Augustin, Sam Ockman und Eric Raymond (später auch Linus Torvalds und Bruce Perens) „Open-Source“.

Damit eine Software Open-Source genannt werden kann, muss sie die 10 Gebote erfüllen, die die Lizenzregelung für Open-Source-Software vorschreibt und die die Verwendung von Open-Source-Software regeln sollen.

1. Freie Weitergabe
2. Quellcode-Verfügbarkeit
3. Abgeleitete Software
4. Unversehrtheit des Quellcodes eines Autors
5. Keine Diskriminierung von Personen oder Gruppen
6. Keine Einschränkungen für bestimmte Anwendungsbereiche
7. Weitergabe der Lizenz
8. Die Lizenz darf nicht auf ein bestimmtes Produktpaket beschränkt sein
9. Die Lizenz darf die Weitergabe zusammen mit anderer Software nicht einschränken
10. Die Lizenz muss technologieneutral sein

Die 10 Gebote sind aus [www.opensource.org/osd.html](http://www.opensource.org/osd.html) entnommen und frei übersetzt. Auf dieser Seite kann eine detaillierte Beschreibung dieser Gebote eingesehen werden.

Aufgrund der fortschreitenden Verbreitung von Open-Source entstanden Diskussionen über die Lizenzregelungen. Daraus haben sich unterschiedliche Lizenzmodelle entwickelt, die für den Anwender und den Entwickler möglichst wenig Einschränkungen in der Verwendung bedeuten und die die Weiterentwicklung von Open-Source-Anwendungen positiv beeinflussen sollen.

### 7.2.2 Lizenzmodelle

#### **Public Domain**

Stellt im Sinne von Open-Source keine Lizenz dar. Es bedeutet, dass jeder damit alles machen kann und der Autor auf jeglichen Anspruch an sein Werk verzichtet.

#### **BSD (Berkeley Software Distribution) License**

Bei dieser Lizenz besteht die Einschränkung, dass in allen veränderten Dateien die ursprünglichen Autoren/Programmierer genannt werden müssen. Ein weiterer Punkt ist, dass der Programmierer von jeglicher Haftung ausgeschlossen ist und die Software für eigene Entwicklungen verwendet werden kann, ohne dass das Ergebnis automatisch dem Open-Source-Modell unterliegen muss.

#### **GNU General Public License (GPL)**

Die Lizenz wurde vor einem politischen Hintergrund definiert. Unterliegt der Sourcecode der GPL-Lizenz, so unterliegt die Weiterentwicklung automatisch der GPL und darf auch nicht mit weiteren Einschränkungen belegt werden. Daraus folgt, dass der Sourcecode nicht für die Basis von kommerzieller Software verwendet werden kann und so für die kommerzielle Weiterentwicklung unbrauchbar ist. Auch schließt die GPL eine Haftung und Gewährleistung aus.

#### **GNU Library General Public License (LGPL)**

Die LGPL entspricht im Wesentlichen der GPL, nur dass die Weiterentwicklungen von Bibliotheken, die der LGPL unterliegen, nur der LGPL und nicht der GPL

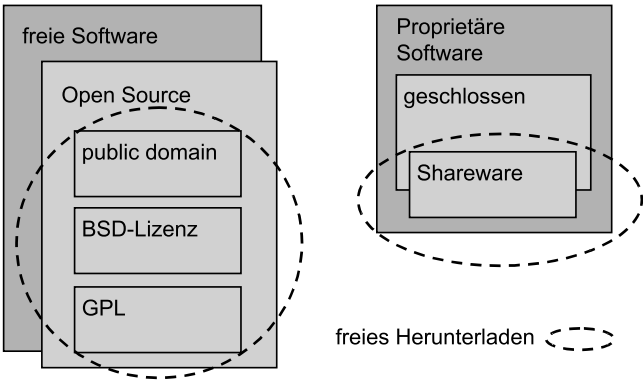
unterliegen müssen. Werden Änderungen in der Bibliothek durchgeführt, unterliegen diese automatisch wieder der LGPL. Beschränkt man sich als Unternehmen auf die Verwendung der durch LGPL geschützten Bibliothek, so ist diese für die kommerzielle Verwendung geeignet.

**QPL, NPL und andere**

Sind meistens Derivate der genannten Modelle und damit den genannten Lizenzen sehr ähnlich und werden selten eingesetzt.

**Tabelle 7.1.** Gegenüberstellung von Lizenzmodellen

	Shareware	Freeware	BSD, NPL	LGPL	GPL
kostenfrei	•	•	•	•	•
frei verwendbar	•	•	•	•	•
Verwendung nicht zeitlich eingeschränkt		•	•	•	•
Sourcecode veröffentlichen			•	•	•
Sourcecode darf verändert werden.			•	•	•
Veränderungen müssen wieder frei sein				•	•
Vermischung von proprietärer Software untersagt					•



**Abbildung 7.3.** Einordnen der Open-Source-Lizenzen

**7.2.3 Open-Source in Verbindung mit Workflow**

Das Angebot von Workflow-Engines auf Basis von Open-Source ist mittlerweile groß und breit gefächert. Es liegen Workflow-Engines vor, die in unterschiedlichen Programmiersprachen und Technologien realisiert sind. Alle Projekte haben

aber gemeinsam, dass der Grad der verwendeten Standards sehr hoch ist. Das bedeutet, dass die Projekte alle in Anlehnung an die WfMC und an die W3C realisiert wurden. Vor dem Einsatz einer solchen Workflow-Engine sollte man sich zuerst einen Überblick über die bestehenden Workflow-Engines verschaffen. Hierzu können zwei Links genannt werden, mit denen man einen Überblick erhält.

[http://www.manageability.org/blog/stuff/workflow\\_in\\_java/view](http://www.manageability.org/blog/stuff/workflow_in_java/view)  
<http://www.sourceforge.net>

Prinzipiell kann gesagt werden, dass Open-Source-Workflow-Engines für die Implementierung in Eigenentwicklungen geeignet sind. Nicht zuletzt deshalb, weil das Angebot von ausgelieferten Modulen für die Administration, Modellierung und Analyse sehr begrenzt ist. Die Eigenständigkeit dieser Projekte ist sehr selten gegeben. In der Regel muss das Projekt durch Eigenentwicklung oder durch die Anbindung weiterer Module, die in einem kommerziellen Produkt vorliegen, komplettiert werden.

Als Beispiel für eine Workflow-Engine kann das *open-source workflow toolkit* (wftk) genannt werden, welches unter <http://www.vivtek.com/wftk/> angeboten wird. Die Seiten enthalten alle Dokumentationen, Sourcecode, Beispiele und die Funktionalitäten, die in Zukunft eingebaut werden sollen.

Da die Projekte einer hohen Flexibilität unterliegen, soll auf eine genaue Beschreibung einzelner Projekte und einer Gegenüberstellung verzichtet werden, da der Zeitraum zwischen dem Schreiben und dem Lesen dieses Buches ausreicht, um in einem Projekt weitere Funktionen zu realisieren und zu veröffentlichen.

Wird in Betracht gezogen, ein Open-Source-Projekt in die Entwicklung mit einzubinden, sollten neben den allgemeinen Evaluierungskriterien die folgenden Kriterien berücksichtigt werden:

- Welchen fachlichen Ursprung hat das Projekt? (DMS, Groupware usw.)  
Daraus lässt sich, wie bei einem kommerziellen Produkt, erkennen, auf was bei der Entwicklung der Schwerpunkt gelegt wird bzw. wurde.
- Wie viele Projektbeteiligte arbeiten an dem Open-Source-Projekt mit?  
Je höher die Anzahl der Beteiligten ist, um so höher ist die Wahrscheinlichkeit, dass das Projekt kurzfristig neue Features erhält bzw. Fehler zeitnah entfernt werden und das Projekt auch in Zukunft gepflegt wird.
- Steht hinter dem Projekt ein Unternehmen oder nur ein loser Verbund von Entwicklern? Ein Unternehmen würde die oben beschriebenen Punkte positiv beeinflussen.

#### 7.2.4 Kommentar

Durch die nur teilweise abgeschlossene Standardisierung wird die Verbreitung von Web Services weiterhin gebremst sein. Ist die Standardisierung aber in ausreichendem Maße abgeschlossen, werden die Produkthersteller für Entwicklungsumgebungen nicht umhin kommen, die Erstellung von Web Services in ihr Produkt mit aufzunehmen.

Prinzipiell stellen Web Services für die Prozesssteuerung eine ideale Basis dar, um die Prozesse wie aus einem Baukastensystem zusammenzustellen und über die Grenzen des Firmennetzwerkes betreiben zu können.

Open-Source wird in den aktuellen Projekten derzeit wenig eingebunden. Dabei kann es gerade für Eigenentwicklungen hilfreich sein, dass Teile der Lösung durch Open-Source-Projekte abgedeckt werden. Die bekanntesten Projekte sind Linux, Apache, Tomcat, jBoss, mySQL und PostgresQL. Die genannten Projekte haben bezüglich ihrer Installierbarkeit, Bedienbarkeit, Dokumentation und der Release-Planung einen hohen Reifegrad erreicht, so dass sie den Vergleich mit kommerziellen Software-Produkten nicht mehr zu scheuen brauchen. Den Vorteil von Open-Source stellt in erster Linie nicht die kostengünstige Implementierung der Projekte dar, sondern vielmehr die Herstellerunabhängigkeit von Software-Produkten, die freie Wahl von Dienstleistern für die Unterstützung der Realisierung und eine einfachere Kooperation durch Standards.



## 8 Werkzeuge

Auf dem Markt ist neben den genannten Open-Source-Projekten eine Vielzahl kommerzieller Produkte vorhanden, die in Verbindung mit Workflow-Management gebracht werden. Jedes dieser Produkte hat seine Stärken und Schwächen und ist auf Basis unterschiedlicher Architekturen realisiert worden. In diesem Kapitel werden einzelne Produkte, die für die Automatisierung der Geschäftsprozesse genutzt werden können, beispielhaft vorgestellt.

### 8.1 Workflow-Management-Systeme

Ziel dieses Kapitels ist es, den Leser bei der Auswahl eines WfMS zu unterstützen. Hierbei wird der Fokus auf drei verschiedene Architekturen gelegt, die durch unterschiedliche Produkte vertreten sind. Für die J2EE-Architektur steht das Produkt CARNOT der Firma CARNOT AG, für eine unter Windows realisierte Lösung das Produkt e-Work der Firma Metastorm Deutschland GmbH, und für ein datenbankbasiertes WfMS das Produkt *EasyFlow*<sup>®</sup> der Firma TOPAS InformationsTechnologien GmbH. Alle drei Produkte haben aufgrund ihrer Architektur und der implementierten Organisationsstruktur, die hinter diesen Produkten steht, unterschiedliche Stärken und Schwächen. Durch die Vorstellung und Beurteilung der drei Produkte entsteht am Ende des Kapitels ein Kriterienkatalog, den der Leser für die Auswahl eines für seine Aufgabenstellung geeigneten Produktes verwenden kann. Es soll noch einmal erwähnt werden, dass dies nicht der Vorgehensweise einer Evaluierung entspricht, in der zuerst die Anforderungen des Unternehmens an ein Produkt zusammengestellt werden müssen. Es hat vielmehr den Hintergrund, dass der Leser diesen Kriterienkatalog besser verstehen und um seine Anforderungen ergänzen kann, wenn dieser anhand von Beispielen erstellt wird.

Zur Unterscheidung von WfMS-Produkten werden die Kriterien in die folgenden vier Kategorien eingeteilt:

- *Unternehmen*, Eckdaten des Unternehmens
- *Prozessarchitektur*, für die das Produkt realisiert wurde
- *Software-Architektur* des Produktes
- *Vor- und Nachteile* des Produktes

Diese Kategorien werden weiter untergliedert und müssen mit den Anforderungen des Unternehmens abgeglichen werden, in dem die Lösung realisiert werden soll.

### 8.1.1 CARNOT

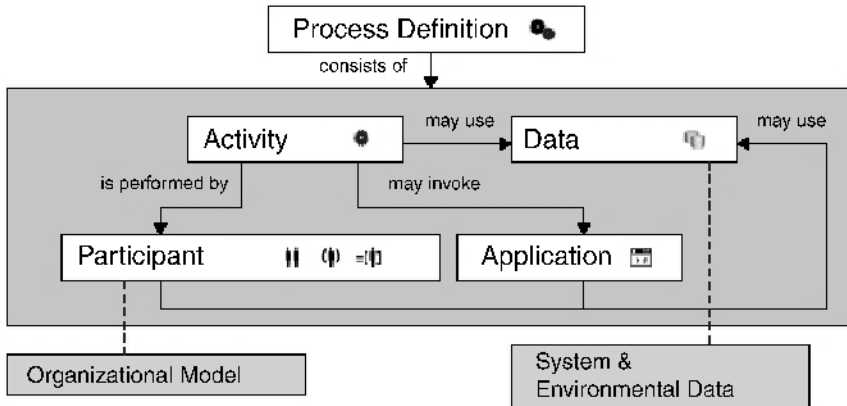
CARNOT ist ein WfMS-Produkt der Firma CARNOT AG ([www.carnot.ag](http://www.carnot.ag)) mit Firmensitz in Deutschland, die 2000 gegründet wurde. Das Produkt CARNOT wurde mit dem Firmenstart angeboten und basiert auf dem J2EE-Standard. Um den Lebenszyklus (Life Cycle) eines Geschäftsprozesses abdecken zu können, bietet CARNOT für die Modellierung den *Process Definition Desktop*, für die Ausführung den *Process Execution Desktop* und für die Administration und Auswertung den *Administration Desktop* an. Durch den J2EE-Standard und die bereits realisierten Schnittstellen kann CARNOT auf unterschiedliche Weise implementiert werden. Es werden drei verschiedene Formen für die Verwendung der Process-Engine angeboten: Ein Standard-Java-Client (*Process Execution Desktop*), ein Web-Client und die Integration in eine Anwendung. Mit dem Modellierungswerkzeug können die Geschäftsprozesse durch bereitgestellte grafische Symbole modelliert und zur direkten Integration von EJB-Komponenten, Host-Systemen und weiteren ERP- und SCM-Systemen verwendet werden. Dieses Verfahren basiert bei CARNOT auf der JCA<sup>12</sup>-Schnittstelle. Für die Darstellung der Geschäftsprozesse, deren Definition in einer Datei im XML-Format gespeichert werden kann, werden unterschiedliche Fenster angeboten. Ein zusätzliches Fenster wird für die Definition des Organisationsmodells bereitgestellt. Die darin definierten Rollen können in der Darstellung der Geschäftsprozesse den Aktivitäten zugewiesen werden. Die bereitgestellten Werkzeuge werden Im Folgenden noch detaillierter beschrieben.

#### Prozessarchitektur

CARNOT wurde in Anlehnung an die Vorgaben der WfMC entwickelt. In der aktuellen Version 2.7 ist die Schnittstelle 1 realisiert, mit der über ein ASCII-Datenformat die Prozessmodelle ausgetauscht werden können. In Abbildung 8.1 wird das Referenzmodell dargestellt, welches die Komponenten des CARNOT-Metamodells enthält. Das in CARNOT verwendete Modell enthält alle Prozesse des Unternehmens. Darunter werden die Prozesshierarchien, die Aktivitäten und Kontrollflussstrukturen, Datenflüsse, Aspekte der Applikationsintegration sowie die modellierten Ressourcen verstanden. Die Prozesshierarchien enthalten die Prozessdefinitionen sowie die verschiedenen Aktivitäten, Regeln und Kontrolldaten, um zur Laufzeit Prozesse steuern zu können. Eine solche Prozessdefinition wird nach dem Abschluss der Modellierung an die Workflow-Engine übergeben, damit diese aufgrund von Geschäftsvorfällen eine Prozessinstanz generieren kann.

---

<sup>12</sup> JCA: Java Connector Architecture von Sun Microsystems



**Abbildung 8.1.** CARNOT-Referenzmodell

In CARNOT ist die Aktivität (Activity) das Kernelement eines Prozesses bzw. einer Prozessinstanz. Aktivitäten können manuell oder automatisch gestartet werden und unterschiedliche Aufgaben erfüllen. Diese Aufgaben können zur Folge haben, dass Applikationen ausgeführt, Subprozesse gestartet oder manuelle Aufgaben durchgeführt und bestätigt werden. Als Besonderheit können die Aktivitäten auch als Synchronisationspunkt für parallele Verläufe dienen. Wird eine Aktivität manuell ausgeführt, so wird diese durch eine Ressource ausgeführt, welche eine Rolle oder eine Organisation sein kann. Die Aktivitäten sind durch Transitionen verbunden, wie sie in dem Kapitel Prozessmodellierung beschrieben wurden. Auch in CARNOT muss eine Transition eine Bedingung erfüllen, um die darauf folgende Aktivität zu starten. Eine solche Bedingung kann das Auslesen von Prozessdaten oder eine Reaktion auf Ereignisse beinhalten. Folgt eine Aktivität mit einem Applikationsaufruf, so ist bei der Modellierung darauf zu achten, dass der Aufruf der Applikation vor der weiteren Bearbeitung der Prozessinstanz beendet werden muss.

Laut CARNOT werden die folgenden Applikationstypen unterstützt:

- SessionBeans: Die Applikation ruft eine Methode einer Stateless oder Stateful SessionBean auf.
- Java-GUI-Komponenten: Die Applikation ist eine Java-GUI, die im *Process Execution Desktop* angezeigt wird.
- JavaServer Pages: Die Applikation ist eine JSP, die im *Web Execution Desktop* angezeigt wird.
- bestehende Anwendung oder Host-Dialog

Für einen definierten Ablauf eines Prozesses sind Datencontainer zur Speicherung von Prozessdaten und Laufzeitdaten erforderlich. Diese werden für Bedingungen der Transitionen oder als Ergebnis für einen Prozess benötigt. CARNOT bietet die Möglichkeit, diese Daten im *Process Definition Desktop* zu definieren und in dem Prozess-Diagramm den Aktivitäten oder den Bedingungen zuzuordnen. Ein sol-

ches Vorgehen erfolgt auch bei der Modellierung von Ressourcen: in einem Diagramm können die hierarchischen Beziehungen zwischen individuellen Ressourcen und Organisationen definiert werden. Diese werden dann den Aktivitäten im Prozess-Diagramm zugeordnet. Zusätzlich bietet CARNOT die Möglichkeit, sogenannte bedingte Ressourcen zu definieren, die erst zur Laufzeit festgelegt werden. Solche bedingten Ressourcen können Rollen oder Organisationen sein, die eine zugeordnete Aktivität ausführen dürfen.

Allgemein kann man sagen, dass mit CARNOT die Geschäftsprozesse mit Symbolen in grafischen Diagrammen definiert werden. Die Symbole stellen nicht nur Modellelemente dar, sondern es können auch konkrete Funktionalitäten hinterlegt werden. Dies können sowohl Anwendungen als auch Datenstrukturen sein. Ist ein solches Modell vollständig definiert, wird es in die Produktivumgebung einge-  
spielt, ohne einen zusätzlichen Entwicklungsaufwand zu generieren.

### Software-Architektur

Da CARNOT komplett in die J2EE-Architektur integriert wurde, können die J2EE-Komponenten für die Entwicklung von Lösungen verwendet werden. Wie in Abbildung 8.2 zu erkennen ist, wurde die Workflow-Engine (*carnot EJB-Engine*) in einem EJB-Container realisiert und kann durch unterschiedliche Übertragungsprotokolle und Komponenten direkt oder indirekt über eine dokumentierte API angesprochen werden. Das *Modeling Desktop*, in dem die Prozesse grafisch modelliert werden, speichert diese Modelle in dem Model Repository. Dieses ist eine XML-Datei. Werden die modellierten Prozesse in die Workflow-Engine über-  
spielt, wird diese XML-Datei über die Workflow-Engine in die *Audit-Trail-Database* importiert. Beim Import von gleichnamigen Prozessen wird automatisch eine neue Version angelegt und die neue Version des Prozesses verwendet.

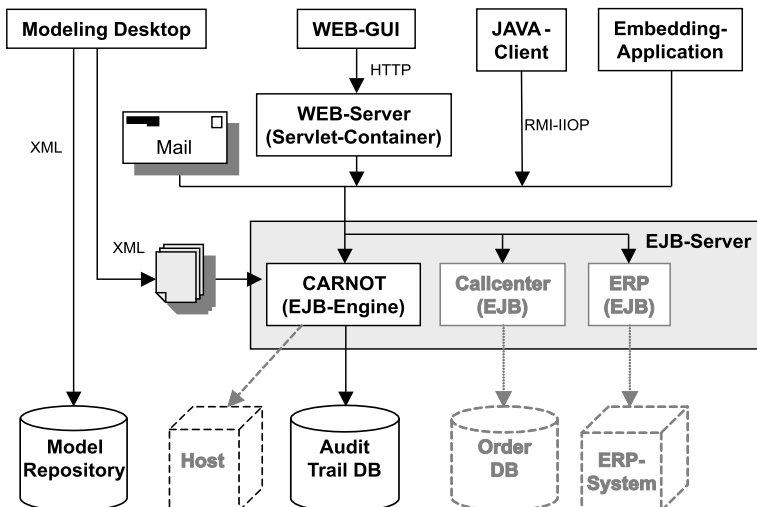


Abbildung 8.2. CARNOT-Architektur

Die Administration und die Auswertung der Prozesse anhand der gestarteten Prozessinstanzen erfolgt durch den *Administration Desktop*. Dieser liest die Laufzeitdaten der Prozessinstanzen aus der Audit-Trail-Database aus. Für die Bearbeitung der Prozessinstanzen werden durch den Produkthersteller drei verschiedene Client-Varianten angeboten. Der *Process Execution Desktop* (local GUI) ist als Java-Client realisiert und kann direkt ohne Anpassungen für einfache Prozesse verwendet werden. In ihm sind die Funktionen enthalten, um alle modellierten Aktivitäten eines Prozesses zu bearbeiten. Zusätzlich wird der *Process Execution Desktop* als Web-GUI angeboten. Dieser HTML-Client ist in einem Browser lauffähig und zeichnet sich durch Betriebssystemunabhängigkeit und Inter-/Intranet-Fähigkeit aus. Die dritte Variante ist die Integration des WfMS in eine Software-Lösung. Hierbei entsprechen die Clients den Anwendungsmodulen, die die Aktivitäten bzw. die Prozessdaten des Geschäftsprozesses bearbeiten. Diese Client-Varianten können innerhalb eines Prozesses wechseln, d.h. ein Prozess kann durch einen Web-Client über das Internet gestartet und durch das *Process Execution Desktop* weiterbearbeitet werden. Durch den J2EE-Standard können weitere Applikationen über die folgenden Komponenten angebunden werden:

### **EJB-Integration**

Aufgrund dessen, dass die Workflow-Engine auf Basis der EJB 2.0-Definition realisiert wurde, kann diese in einem Standard-konformen EJB-Container verwendet werden. Es ist somit auch möglich, dass beliebige SessionBeans für die Ausführung von Aktivitäten genutzt werden können und bereitgestellte EntityBeans sich als transaktionale und persistente Prozessdaten in Prozessen verwenden lassen. In der Abbildung 8.2 sind im EJB-Container bereits weitere mögliche kundenspezifische Komponenten wie ein ERP- und ein Callcenter-System integriert. Dies soll verdeutlichen, dass eine Integration solcher Lösungen mit ihren eigenen Datenhaltungen und Funktionalitäten möglich ist.

### **Java Mail**

Durch die Anbindung eines Mailservers über Java Mail können z.B. Prozesse durch eingehende Mails gestartet oder Mails aufgrund von Ereignissen versendet werden.

### **Java Transaction Service /Java Transaction API (JTS/JTA)**

Durch JTS/JTA werden neben den Prozessdaten auch die Laufzeitdaten (Logging-Daten) der Prozessinstanzen in der Audit-Trail-Datenbank gespeichert. Um die Laufzeitdaten später effektiv auswerten zu können, werden diese synchron zu den Prozessdaten gespeichert.

### **Java Message Service (JMS)**

Für die Abwicklung von nebenläufigen Prozessen wird für die Nachrichtenübermittlung JMS eingesetzt, welches auch den externen Prozessesstart ermöglicht.

### **Import/Export**

Durch eine Import- und Export-Schnittstelle können die Prozessmodelle im XML-Format ausgetauscht werden. Diese Schnittstelle wird zukünftig auch für die An-

bindung des ARIS-Toolsets verwendet, um die Prozessmodelle, die in ARIS modelliert wurden, übernehmen zu können.

### Java Database Connectivity (JDBC)

CARNOT bindet Datenbanken über JDBC an. Als Basis für die Audit-Trail-Datenbank kann eine JDBC 2.0-konforme Datenbank verwendet werden. Von CARNOT wurden für die Produkte von Oracle, IBM DB/2 und PostgreSQL bereits entsprechende Tests und Installationen durchgeführt.

Der *Process Definition Desktop* wurde in Java realisiert und kann dadurch betriebssystemunabhängig betrieben werden. Mit diesem Desktop werden die Prozessmodelle erstellt, welche sämtliche Informationen über Anwendungskomponenten und Organisationsstruktur enthalten. Bevor das Modell in die Laufzeitumgebung überspielt werden kann, wird im Desktop kontrolliert, ob das Modell konsistent ist. Der Modellierer wird auf eventuelle Fehler hingewiesen.

Es ist zu beachten, dass immer nur ein Modell pro Unternehmen verwaltet werden kann. Es werden in diesem Modell alle Prozesse und das Organigramm in den unterschiedlichen Ausprägungen verwaltet. Die Navigation durch das Modell erfolgt im linken Bereich des Desktops, in dem durch ein Kontextmenü (rechte Maustaste) weitere Elemente des Modells wie Prozesse, Personen, Applikationen und Daten angelegt werden können. Auf der rechten Seite können diese dann grafisch in den Diagrammen bearbeitet und untereinander verknüpft werden. Bevor mit der Modellierung begonnen werden kann, muss ein Benutzer mit den entsprechenden Rechten unter Verwendung des *Administration Desktop* angelegt werden. Durch die Benutzerauthentifikation wird der Benutzerkreis für das Bearbeiten der Pro-

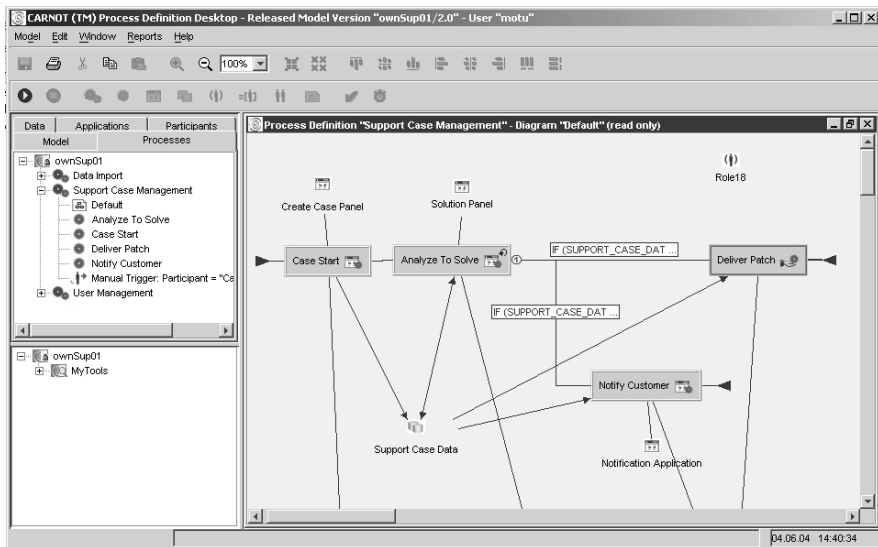


Abbildung 8.3. CARNOT Definition Desktop

zessmodelle eingeschränkt und ermöglicht so ein kontrolliertes Bearbeiten an den unterschiedlichen Prozessen.

Der *Process Execution Desktop* kann in drei verschiedenen Varianten verwendet werden. Zum einen als ein in Java realisierter Client, mit dem der Anwender alle Prozessaktivitäten ausführen kann, die eine Benutzereingabe erwarten. Bevor der Anwender den Client nutzen kann, muss er sich durch eine Benutzerauthentifikation an CARNOT anmelden. Die Workflow-Engine ermittelt anhand der Anwenderkennung die zugewiesenen Rollen, die Prozessaktivitäten und die Prozesse, die bearbeitet bzw. gestartet werden können. Ist der Anwender berechtigt, mindestens einen Prozess zu starten oder zu bearbeiten, erhält er auf der rechten Seite eine Prozessliste, in der die Prozesse aufgelistet sind, die er starten darf, und eine Worklist (Tätigkeitsliste), in der die Aktivitäten aufgelistet sind, die er bearbeiten darf. Auf der linken Seite werden die Daten für die zu bearbeitenden Aktivitäten dargestellt. Ist die Bearbeitung einer Aktivität abgeschlossen oder muss die Bearbeitung abgebrochen werden, kann dies über die in der rechten Menüleiste angezeigten Symbole (grüner Haken, rotes Kreuz) durchgeführt werden.

Zum anderen steht der Browser-basierte Client (*Web Execution Desktop*) zur Verfügung, mit dem der volle Funktionsumfang des Java-Clients für eine Intra- und Internet-Lösung angeboten wird. Beide Varianten können an kundenspezifische Oberflächen angepasst werden.

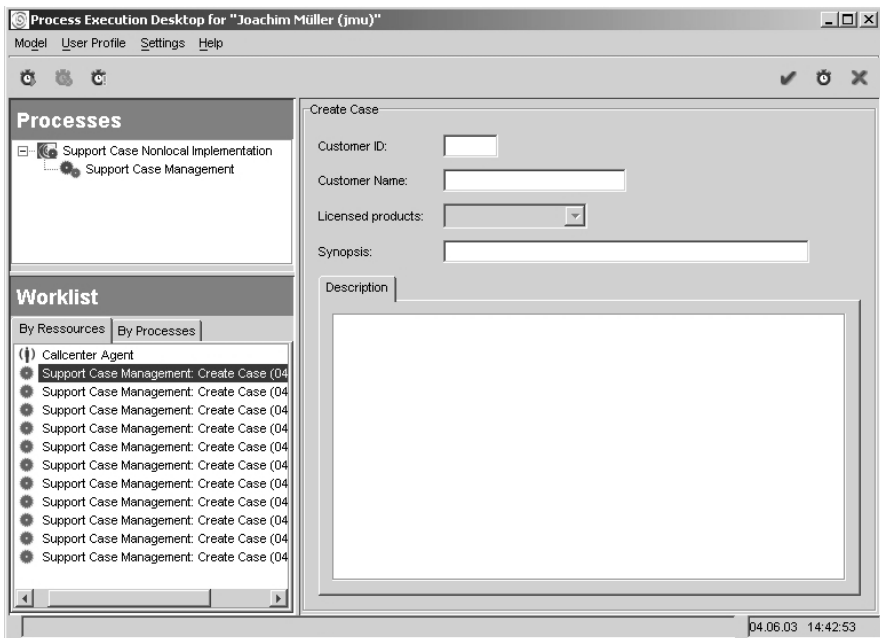


Abbildung 8.4. CARNOT Execution Desktop

Eine dritte Variante des *Execution Desktop* ist der WAP<sup>13</sup>-Client. Mit ihm können Clients auf mobilen Endgeräten wie Handys und PDAs realisiert werden. Dadurch kann der Anwender sich an das WfMS anmelden, Prozesse starten, Aufgabenlisten ansehen und Aktivitäten starten oder beenden. Die WAP-Schnittstelle wurde durch JSPs und WML (Wireless Markup Language) realisiert. WML ist das WAP-Pendant zu HTML und ist eine Abfolge von sogenannten WAP-Decks mit der gleichen Funktionalität wie der *Process Execution Desktop* und der *Web Execution Desktop*. Der WAP-Client kann beliebig angepasst werden: Die Entwickler können die dokumentierte Programmierschnittstelle verwenden und eigene WML-Decks erstellen. Es sind aber wie mit jedem WAP-Client Überlegungen anzustellen, welche Aktivitäten durch das mobile Endgerät mit dem WAP-Client überhaupt sinnvoll durchgeführt werden.

Mit dem *Administration Desktop* können die Prozesse und die Funktionen der Workflow-Engine überwacht und verwaltet werden. Durch sechs unterschiedliche Ansichten können die verschiedenen Attributgruppen wie die Workflow-Engine, die Prozesse, die Aktivitäten, die Anwender und die angebundenen Ereignis-Generatoren (Mail-Trigger, Timer) verwaltet werden.

Bei der Sicht der Workflow-Engine werden Informationen über die gestarteten Prozessinstanzen und deren Zustände dargestellt. Dabei kann man erkennen, wie viele Prozessinstanzen durch die Engine oder durch die Anwender gerade bearbeitet werden, welche sich in einem aktiven Zustand befinden, welche Aktivitäten der Prozessinstanzen gerade gestartet wurden usw.

In der Prozess-Ansicht können Auswertungen mittels Filter über einen Zeitbereich, über die Anwender, über den Prozessstatus und über den Zustand der einzelnen Prozessinstanzen durchgeführt werden. Dies ist detailliert bis auf Aktivitätenebene möglich.

Die Auswertung in der Aktivitäten-Ansicht ermöglicht die Informationsbeschaffung über Flaschenhälse in den Prozessverläufen.

Die Anwender-Ansicht schließlich liefert alle Informationen über Anwender (Akteure). Es können Anwenderstatistiken, allgemeine Benutzerinformationen einschließlich Anwender-ID, Name, Konto, E-Mail und Beschreibung sowie komplette Informationen über die durchgeführten Tätigkeiten und deren Prozessinstanzen erstellt werden. Mit der Ansicht „Daemons“ können die Hintergrundsystemprozesse gestartet und deren Zustand überwacht werden. Durch die Programmierschnittstelle können neben den bereits ausgelieferten Daemons weitere realisiert und mit Hilfe des Desktops verwaltet werden.

---

<sup>13</sup> Wireless Application Protocol: Übertragungsstandard für drahtlose Kommunikation



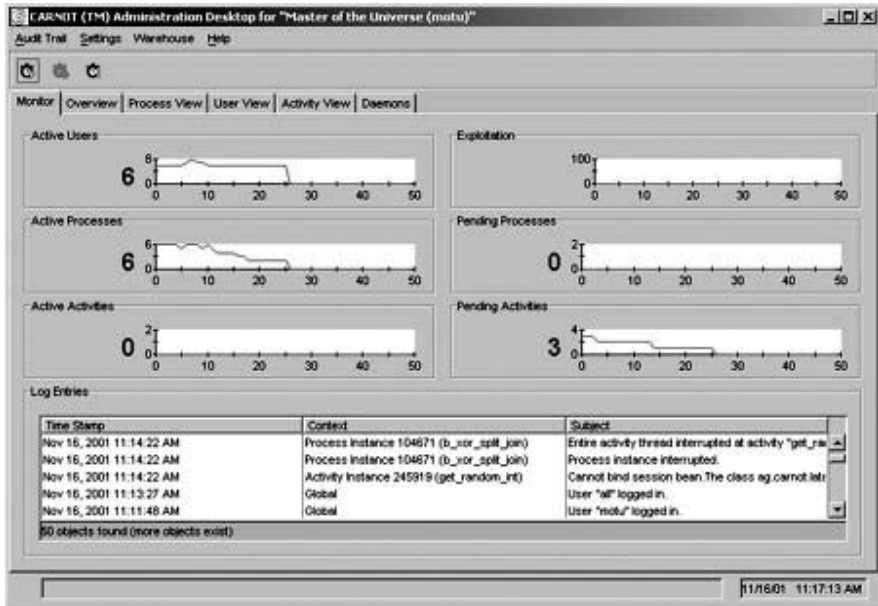


Abbildung 8.5. CARNOT Administration Desktop

## Systemanforderungen

Client:

- Pentium II
- 256 MB RAM
- 40 MB Festplattenplatz
- LAN-Verbindung zum CARNOT-Server mit der Datenbank
- Windows NT/Windows 2000
- JDK 1.3.x

Folgende Applikationsserver können garantiert als Laufzeitumgebung verwendet werden:

- BEA WebLogic 6.1 SP1
- Borland Enterprise Server 5.1
- JBoss 3.2.0
- IBM WebSphere 5.0
- Oracle 9i AS
- Pramati Server 3.0

Folgende Datenbank-Server sind möglich:

- ORACLE 8.1.7
- DB2 7.2
- PostgresQL

Die folgende Beschreibung der Vor- und Nachteile ist nicht nach den oben dargestellten Kategorien sortiert, sondern in der Reihenfolge ihrer Bedeutung.

#### Vorteile von CARNOT

- zukunftsorientierte Architektur durch J2EE-Standard und Anbindung an .NET
- gute Skalierbarkeit der Anwendung
- Erreichbarkeit eines hohen Integrationsgrades
- Anbindung fast aller Komponenten auf Basis des J2EE-Standards möglich
- Simulation des Prozessablaufs (Debugging) im *Definition Desktop*
- Datenaustausch auf Basis von XML
- Anbindung von Open-Source-Projekten wie JBoss und PostgreSQL möglich

#### Nachteile von CARNOT

- Versionisierung der Prozesse kann nicht in die Zukunft (z.B. zum Jahreswechsel) erfolgen. Dadurch muss eine Versionsänderung immer manuell durchgeführt werden und bereits gestartete Prozesse können nicht in der alten Version zu Ende geführt werden.
- Umfangreiche Realisierung eigener Lösungen aufgrund des J2EE-Standards.
- Neben den CARNOT-Lizenzkosten kommen noch Lizenzkosten für die relationale Datenbank und für einen Applikationsserver hinzu (außer bei der Verwendung von jBoss und PostgreSQL).
- Durch das Produkt wird keine Trennung der Entwicklungs-, Test- und Produktionsumgebung vorgegeben. Dadurch muss ein Entwicklungsprozess in der Erstellungs- und Wartungsphase durch das Projektteam implementiert werden.

### 8.1.2 e-Work

e-Work ist ein Produkt der Firma Metastorm Deutschland GmbH ([www.metastorm.com](http://www.metastorm.com)), die 1996 mit dem Unternehmenssitz in Maryland, USA gegründet wurde. In Europa bestehen Niederlassungen in Frankfurt, Zürich und London, wo auch die Produktentwicklung ansässig ist. Weltweit hat Metastorm mehr als 400 Kunden aller Größenordnungen aus unterschiedlichen Branchen.

In dem Produktpaket e-Work sind neben der Workflow-Engine (*e-Work Process Engine*) weitere Werkzeuge für das Business Process Management enthalten. Diese decken die Bereiche der Modellierung durch den grafischen e-Work-Designer für Prozesse und Forms (Dialoge), die Administration durch den e-Work Analyser und den e-Work Administration Manager und die diversen Wizards, die für die

Erstellung von Schnittstellen verwendet werden können, ab. Die auf Basis von Microsoft-Windows-Betriebssystemen realisierte Workflow-Engine und die vorhandenen Schnittstellen (API und XML) erlauben eine Anbindung an den plattformunabhängigen J2EE-Standard und an die eingeschränkt plattformunabhängige .NET-Technologie. Es sind dadurch Lösungen sowohl im Client/Server-Umfeld als auch im Inter-/Intranet-Umfeld realisierbar.

### Prozessarchitektur

In der Prozessarchitektur wird mit dem grafischen Prozess-Designer die *Procedure* als Hauptkomponente erstellt. Eine *Procedure* beinhaltet mindestens eine *Map*<sup>14</sup> und kann mehrere *Forms* (Dialoge) enthalten. Mit jedem Prozessstart wird eine Prozessinstanz erzeugt, welche in e-Work *Folder* genannt wird. Der *Folder* enthält alle relevanten Prozessinformationen für diese Prozessinstanz und wird durch die e-Work Engine von einem Anwender bzw. einer Rolle aufgrund der in einem Prozess modellierten *Actions* zum nächsten Bearbeiter weitergeleitet. Eine *Map* ist ein Rahmen, in dem ein Prozess mit allen Aktivitäten, allen Bedingungen und allen Schnittstellen in der *Procedure*-Datei gesichert wird. Die in einer *Map* enthaltenen Elemente werden *Stages* und *Actions* genannt. Eine *Stage* repräsentiert eine bestimmte Bearbeitungsstufe innerhalb eines Prozesses, d.h. der *Folder* erreicht einen Anwender, der die Aktivitäten bearbeitet, bevor der *Folder* durch die e-Work Engine in die nächste *Stage* verschoben wird. Eine *Action* repräsentiert eine Aktivität innerhalb eines Prozesses. Sie ist notwendig, um einen *Folder* von einem Status (*Stage*) zum nächsten zu schieben. Beispiele für e-Work *Actions* können „Kreditdaten eingeben“, „Genehmigung“ oder „Anwendung starten“ sein.

Die *Forms* sind elektronische Formulare, die von e-Work verwendet werden, um prozessrelevante Daten anzuzeigen und dem Anwender die Möglichkeit zu geben, diese zu bearbeiten. Die *Forms* werden in dem e-Work Designer den *Actions* zugeordnet und können zum einen vom Benutzer ohne jeden Programmieraufwand über die grafische Benutzeroberfläche des e-Work Designers erstellt und einzelnen Prozessaktivitäten zugeordnet werden. Zum anderen kann e-Work aber auch Formulare verwenden, die mit einem Tool eines Drittanbieters erstellt wurden.

### Software-Architektur

e-Work basiert auf Microsoft-Betriebssystemen und bietet Anbindungsmöglichkeiten über Standards wie ODBC, SMTP, MAPI, LDAP, XML sowie die Integration von Microsoft Outlook Client, Novell Groupwise Client und Microsoft Word an. Die Anbindungen werden idealerweise durch das Microsoft Developer Studio und das von Metastorm ausgelieferte Software Developers Kit (SDK) realisiert. Durch das SDK und den e-Work XML Universal Adaptor können Applikationen wie SAP oder Peoplesoft für den Austausch von Daten zwischen e-Work und weiteren Anwendungen verwendet werden.

<sup>14</sup> Eine *Map* enthält den Ablauf des Prozesses.

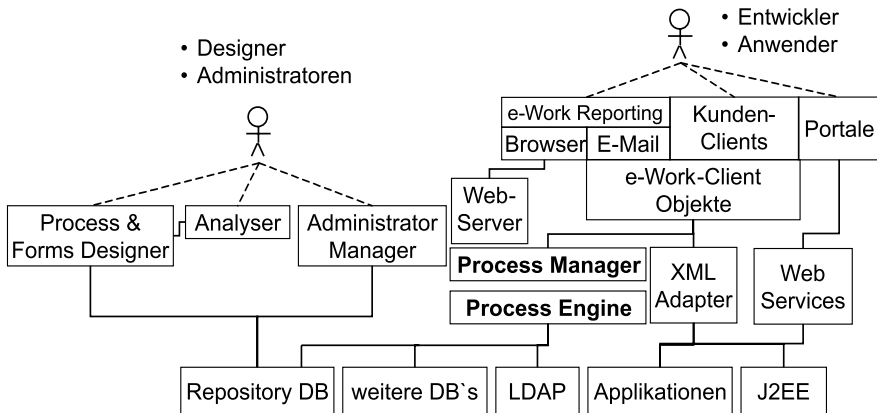


Abbildung 8.6. e-Work-Architektur

Für die Integration stehen dem e-Work-Anwender die oben genannten standardisierten Schnittstellen zur Verfügung, die zum einen auf Produktschnittstellen und zum anderen auf etablierten Standards basieren. Zu diesen Schnittstellen zählen die E-Mail-, Microsoft Word- und Datenbankintegrationen, aber auch Web Services und Microsoft Visual Basic Scripting Edition (VB-Script) oder Microsoft JScript. Unter Integration versteht man, dass Metastorm Plug-Ins für einen E-Mail-Client (Microsoft Outlook 2000 oder Novell GroupWise 5.5) und für Microsoft Word bereits realisiert hat und diese im Produktportfolio enthalten sind. Es ist somit möglich, ohne großen Aufwand die e-Work-Funktionalität in einen Outlook-Client zu integrieren. Für die Anbindung an ein LDAP-Directory bietet e-Work neben dem Microsoft Active Directory noch das Novell eDirectory und die vorhandenen Directories der angebotenen Datenbanken als Alternative an. Als weitere Integrationsmöglichkeit kann über den Microsoft Biztalk Server Adaptor *e-Work* über die .NET-Technologie in bestehende Anwendungen integriert werden.

Als einziges der drei vorgestellten WfMS bietet Metastorm in Zusammenarbeit mit dem Microsoft SharePoint Server eine Funktionalität für die Synchronisation von Prozessen an, die es ermöglicht, den e-Work-Client sowohl online, also mit einer Verbindung zum Server, als auch offline zu betreiben.

Der Lieferumfang von e-Work beinhaltet die folgenden Tools:

### e-Work Engine

Die e-Work Engine ist die Prozesssteuerung für die im e-Work Designer modellierten Prozesse und stellt die Verbindung zur e-Work-Datenbank dar. Die Engine beinhaltet alle Funktionen, die für die Prozesssteuerung mit dem Client benötigt werden. An ihr werden alle externen Programme anhand der Schnittstellen angebunden, die in einen Prozess integriert werden sollen.

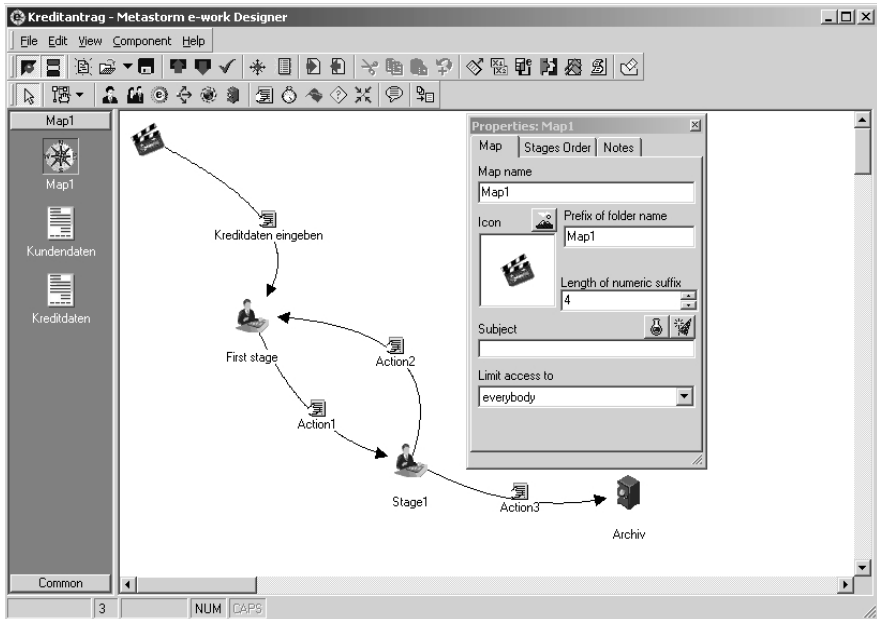


Abbildung 8.7. e-Work Designer

### e-Work Process Manager

Der *e-Work Process Manager* verwaltet die Prozessinstanzen der *Map*, d.h. er führt den aktuellen Zustand der einzelnen Prozessinstanzen und stellt diese über den Client dem Anwender zur Verfügung. Die Verwaltung sagt dem *e-Work Proc*

### e-Work Designer

Mit dem *e-Work Designer* können durch den Modellierer Workflowmodelle grafisch erstellt und Business-Rules (Ablaufbedingungen) zwischen den Aktivitäten definiert werden. Weiterhin können durch den *e-Work Designer* grafisch Dialogmasken (*Forms*) definiert und den Aktivitäten zugeordnet werden.

Der *e-Work Designer* wird auf einem Windows-Client (s. Systemanforderungen) und im ersten Schritt ohne eine Verbindung zur *e-Work Engine* betrieben. Somit ist es beispielsweise möglich, auf einem Laptop mit einer Access-Datenbank die Modellierung zu beginnen. Die Struktur eines Workflowmodells in *e-Work* ist aus den Komponenten *Procedure*, *Map*, *Form*, *Stage* und *Action* aufgebaut, welche im Abschnitt *Prozessarchitektur* beschrieben wurden. Der *e-Work Designer* beinhaltet auf der linken Seite eine Übersicht der bereits verwendeten Komponenten des Prozesses. Diese Komponenten werden auf der rechten Seite in einem separaten Dialog dargestellt und können darin bearbeitet werden. Wird die *Map* auf der linken Seite ausgewählt, so wird der Ablauf des Prozesses mit den eingebundenen Komponenten dargestellt. Über das Kontextmenü (rechte Maustaste) erhält man die Möglichkeit, die Eigenschaften der Komponenten zu bearbeiten. Über diese

Eigenschaften können Name, Typ, Art und Rollen der Komponente zugewiesen und ggf. Visual-Basic-Skripte, je nach Komponente, hinterlegt werden. Über diese VB-Skripte können weitere Funktionen wie z.B. das Auslesen oder Speichern von Daten realisiert werden. Ist die Prozessmodellierung abgeschlossen und in einer Datei gespeichert, kann das Modell bezüglich der Konsistenz automatisch getestet werden. Danach kann durch den *e-Work Designer* der Prozess zu der *e-Work Engine* übergeleitet werden. Im Falle einer bestehenden gleichnamigen Prozessstruktur wird der Anwender gefragt, ob er eine neue Version anlegen oder die bestehende überschreiben will.

### **Users & Roles**

Mit diesem e-Work-Tool können die e-Work-Anwender definiert und Benutzerrollen zugeordnet werden. Diese Rollen müssen zuvor im *e-Work Designer* definiert werden. Die Rollen können dann in Workflowmodellen den *Actions* und den *Stages* zugeordnet werden. Definierte und zugeordnete Rollen werden erst mit der Aktivierung der entsprechenden *Procedure* in der Workflow-Engine aktiv.

### **Administrator**

Mit dem e-work Tool *Administrator* können *Procedures* in den unterschiedlichen Versionen, Informationen der *Maps*, deren *Folders* und Variablen dargestellt und administriert werden. In dem Administrator-Tool besteht weiterhin die Möglichkeit die Designer-Log-Datei auszulesen, in der alle aufgezeichneten Aktionen nach Zeit und *Procedure* geordnet dargestellt werden. Zusätzlich ist in dem Produktpaket noch ein Werkzeug *System Administrator* enthalten. Mit diesem Werkzeug können alle Systemeinstellungen kontrolliert und bearbeitet werden.

### **Web-Client**

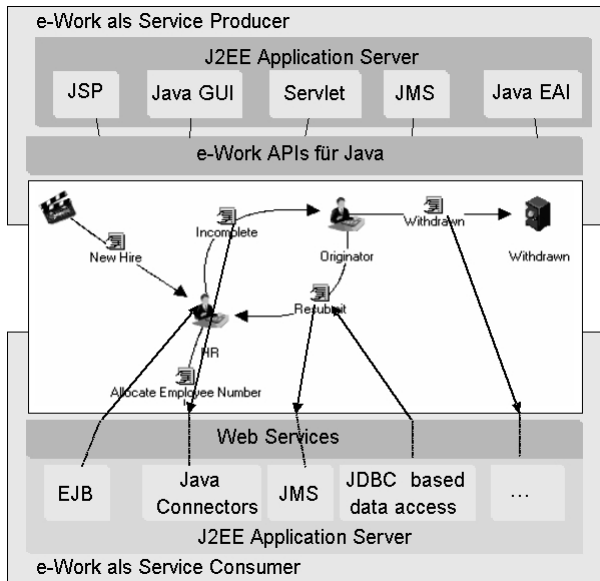
Der Anwender hat mit dem *e-Work Web-Client* über das Inter- oder Intranet den Zugriff auf die *e-Work Process Engine*. Da der Client browserbasiert ist, wird für dessen Verwendung neben einem Web-Browser keine zusätzliche Software auf dem Client-Rechner benötigt. Der Anwender meldet sich mit seiner Kennung und seinem Passwort über einen Login-Dialog an. Die Zugangsdaten wurden zuvor durch das Überleiten des Prozesses (*Procedure*) und die Verknüpfung der Anwenderkennung mit dem Prozess im *User&Roles*-Tool eingerichtet. Hat sich ein Anwender an die *e-work Engine* angemeldet, stehen ihm seine aktuelle Aufgabenliste (*toDo-list*), die Statusliste (*watch-list*) oder leere Formulare (*blank forms*) zur Verfügung, um die ihm zugeordneten Aktivitäten durchführen zu können.

Die Aufgabenliste enthält die *folder*, die der Anwender im Rahmen eines Prozesses bearbeiten muss. Über die Statusliste, welche den *Folder* enthält, kann der Anwender den Status der Prozesse einsehen, die er zuvor bearbeitet hat, die aber noch nicht beendet sind. Der Anwender kann so erkennen, in welchem Bearbeitungszustand sich „seine“ Prozesse befinden. Unter den leeren Formularen sind all jene Prozesse angeordnet, die der Anwender aufgrund seiner Rollenzuweisung starten darf. Der Anwender wählt einen Prozess aus und startet diesen. Die neu gestartete Prozessinstanz (*Folder*) wird allen Anwendern, die die modellierte Rolle inne haben, in die *toDo-list* eingestellt. Stimmt die zugeordnete Rolle der ersten

Aktivität mit einer der Rollen des Prozessstarters überein, hat dieser die Möglichkeit, die Prozessinstanz direkt zu bearbeiten.

Für die Anbindung an Datenbanken werden produktspezifische ODBC-Treiber eingesetzt, die auf jedem Client installiert werden müssen, auf dem der *Designer* und die Tools installiert sind. Der Web-Client benötigt diese nicht.

Neben den oben genannten Standards bietet e-Work auch die Möglichkeit, via Web Services mit Anwendungen von Drittanbietern zu kommunizieren. Web Services ermöglichen einem e-Work-Prozess, mit einer Anwendung wechselseitig über das Internet zu kommunizieren. Durch die XML-basierte Architektur von e-Work können e-Work-Prozesse als Web Services angeboten werden und von Toolkits wie .NET und dem IBM Web Services-Development Toolkit aufgerufen werden. Es besteht sowohl die Möglichkeit, dass der Prozess Web Services (als consumer) verwendet, als auch, dass der Prozess Web Services (als producer) für eine Anwendung zur Verfügung stellt.



**Abbildung 8.8.** e-Work-Architektur Web Services

### Systemanforderungen

Die Hard- und Software-Anforderungen sind abhängig davon, welche e-Work-Module (Engine, Designer usw.) installiert werden sollen. Die hier gemachten Angaben basieren auf der Version 5.3.

#### Prozess-Server

- Windows 2000 (SP2), Windows NT 4.0 (SP6a), Windows XP, MS Transaction Server, MS Distributed Transaction Coordinator
- 256 MB RAM min.
- 6,5 MB Festplattenplatz + ODBC

#### Designer

- Windows 98, Windows 2000 (SP2), Windows NT4.0 (SP6a), Windows XP
- 64 MB RAM
- 3,9 MB Festplattenplatz + ODBC + ADO<sup>15</sup>

#### Users & Roles

- Windows 98, Windows 2000 (SP2), Windows NT4.0 (SP6a), Windows XP
- 32 MB RAM
- 2,3 MB Festplattenplatz + ODBC + ADO

#### Administrator

- Windows 98, Windows 2000 (SP2), Windows NT4.0 (SP6a), Windows XP
- 32 MB RAM
- 2,2 MB Festplattenplatz + ODBC + ADO

#### Datenbankprodukte (werden alle über ODBC-Treiber angebunden)

- MS SQL Server 7 (SP3), MS SQL Server 2000
- ORACLE 8.1.7 (Win NT), ORACLE 8.1.6 (Win NT), ORACLE 8.1.5, ORACLE 9i
- Microsoft Access 2000

#### Web-Server

- MS Internet Information Server 4 SP5 (Win NT)
- MS Internet Information Server 5 SP1 (Win 2000)

#### Web-Browser

- MS Internet Explorer 5.01 (SP2)
- MS Internet Explorer 5.5 (SP1 oder SP2)
- MS Internet Explorer 6.0
- Netscape 4.75 (unterstützt keine *e-work-Forms*, sondern nur extern generierte Formulare)

Die Vor- und Nachteile sind nicht nach den oben genannten Kategorien sortiert, sondern in der Reihenfolge ihrer Bedeutung.

---

<sup>15</sup> ActiveX Data Objects: Über ADO lassen sich fast alle geläufigen Datenbanken zugreifen, ob Access, SQL-Server, Oracle, Informix oder weitere Datenbanksysteme.



### Vorteile von e-Work

- effektives Bearbeiten eines Prozesses von der Modellierung bis zum Betreiben des Prozesses
- hoher Integrationsgrad bzgl. Mail-Clients wie Microsoft Outlook 2000 oder Novell GroupWise 5.5 und allgemein bzgl. Microsoft-Produkten wie z.B. Word oder Excel
- umfangreiche Schnittstellen-Architektur
- Verwendung von Web Services
- Erstellen der Prozessdokumentation per Mausklick
- Es können für den Zweck der Modellierung der *Designer* und die Tools mit einer Access-Datenbank betrieben werden.

### Nachteile von e-Work

- Versionisierung der Prozesse kann nicht in die Zukunft (z.B. zum Jahreswechsel) erfolgen. Dadurch muss eine Versionsänderung immer manuell durchgeführt werden und bereits gestartete Prozesse können nicht in der alten Version zu Ende geführt werden.
- Es ist immer ein Server mit einem Windows-Betriebssystem erforderlich.
- Trennung der Produktions-, Test- und Entwicklungsumgebung werden nicht durch e-Work vorgegeben
- Client kann nicht anwenderabhängig konfiguriert werden
- keine grafische Darstellung der Prozesshistorie

#### 8.1.3 EasyFlow

*EasyFlow* ist ein Produkt der Firma TOPAS InformationsTechnologien GmbH (kurz TOPAS-IT GmbH - [www.topas-it.de](http://www.topas-it.de), [www.easyflow.de](http://www.easyflow.de)), die 1993 gegründet wurde. Das Produkt wurde 1996 in der Version 1.0 bei einem Finanzdienstleister implementiert, der mit derzeit ca. 1.000 Anwendern die größte Installation von *EasyFlow* besitzt.

Die technologische Basis von *EasyFlow* ist eine relationale Datenbank, in der die Workflow-Funktionalität durch interne Funktionen realisiert wurde. Bisher wurden die Installationen mit Oracle ab Version 7 durchgeführt, da diese die Anforderungen von *EasyFlow* durch die Funktionalität der Stored Procedures am besten abdeckt. Die Firma TOPAS-IT GmbH bietet für die ca. 100 Stored Procedures ein SQL Application Programming Interface (API) direkt in der Datenbank oder in Form von Application Programming Libraries (APL) für SQL Windows32 und in Form von DLLs für Programmiersprachen in den Windows-Umgebungen. Diese APIs ermöglichen die Integration in bestehende Anwendungen und können so die Prozesse durch *EasyFlow*-Standarddialoge oder durch die Eigenentwicklung von Dialogen steuern. Die Einsatzmöglichkeit von *EasyFlow* in bestehenden heterogenen IT-Umgebungen war ein wichtiges Kriterium bei dessen Entwicklung. Das

Prozessmanagement wird durch ein Definitions-, Organisations- sowie ein Simulations- und Administrationswerkzeug unterstützt.

### Prozessarchitektur

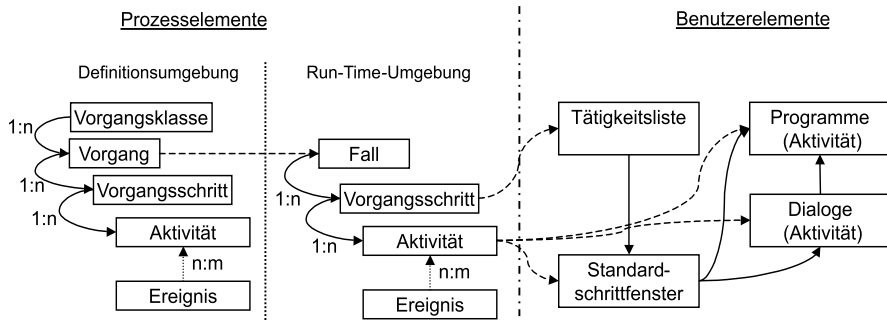
Für die Definition, den Test, die Verwaltung und das Betreiben der Geschäftsprozesse werden durch den Produkthersteller drei verschiedene Umgebungen bereitgestellt. Die Entwicklungs-, Test- und Produktionsumgebungen sind technisch voneinander getrennt und müssen von jedem Geschäftsprozess für dessen Implementierung durchlaufen werden. Durch diese Vorgehensweise kann ein hohes Maß an Qualität während der Definition der Prozesse erreicht werden. In der Entwicklungsumgebung steht das dialogbasierte Definitionswerkzeug zur Verfügung, mit dem der Ablauf und die Anbindung der Fachanwendungen für den Geschäftsprozess definiert wird. Wie oben bereits erwähnt stellt *EasyFlow* keine eigene grafische Modellierungsoberfläche zur Verfügung – es werden alle Prozessbestandteile (Vorgänge, Schritte, Aktivitäten und Ereignisse) über Dialoge in das System eingegeben. Eine weitere Möglichkeit, Geschäftsprozesse zu modellieren, besteht über die Anbindung des Modellierungswerkzeuges ADONIS von BOC. In ADONIS können die Geschäftsprozesse grafisch modelliert werden und in eine XML-ähnliche Datei exportiert werden. Ein Geschäftsprozess kann durch diese Datei in die Entwicklungsumgebung von *EasyFlow* importiert und dort mit *EasyFlow*-spezifischen Anpassungen ergänzt werden. Danach wird der Geschäftsprozess in die Testumgebung überspielt und entweder durch das Simulationswerkzeug oder mit den bereits realisierten Anwendungskomponenten getestet. Sind die Tests positiv verlaufen, wird die Version des Geschäftsprozesses mit einem Gültigkeitsdatum in die Produktion überspielt. Wird das Gültigkeitsdatum erreicht, zieht das WfMS die Vorgängerversion des Geschäftsprozesses zurück und stellt automatisch die neue Version zur Verfügung. Es werden nur neu gestartete Geschäftsprozesse mit der neuen Version gestartet. Bereits zuvor gestartete Geschäftsprozesse werden in der alten Version zu Ende geführt.

Bevor auf die Workflow-Objekte von *EasyFlow* eingegangen wird, werden die in *EasyFlow* verwendeten Begriffe der WfMC-Terminologie gegenübergestellt.

**Tabelle 8.1.** Gegenüberstellung der Terminologie

Deutsche Bezeichnungen	EasyFlow	WfMC-Terminologie
	Vorgangsklasse	-
Vorgang, Vorgangstyp	Vorgang	Business Process
Schritt, Aktivität	Vorgangsschritt	Activity
Arbeitsschritt, Aktivität, Tätigkeit	Aktivität	Work Item
Postkorb, Arbeitskorb	Tätigkeitsliste	Activity Instances List
	Aktivitätenliste	Work List
Geschäftsvorfall	Fall	Process Instance

In der von *EasyFlow* verwendeten Terminologie wird der Vorgang in Vorgangsschritte und diese wiederum in einzelne Aktivitäten gegliedert. Verschiedene Vorgänge, die sich aufgrund der Ähnlichkeit im Ablauf gruppieren lassen, werden in einer Vorgangsklasse zusammengefasst. Es besteht dabei die Möglichkeit, Abweichungen innerhalb einer Vorgangsklasse zu modellieren. Die Workflow-Objekte lassen sich in einer hierarchischen Darstellung wie in Abbildung 8.9 anordnen.



**Abbildung 8.9.** *EasyFlow*-Beziehungen

Der Vorgang fasst einen Geschäftsprozess fachlich zusammen. Ihm sind Standardvariablen zugeordnet, die während des laufenden Falles belegt und zur Ablaufsteuerung verwendet werden können. Innerhalb der Hierarchie stellt der Vorgangsschritt mit seinen Aktivitäten das wesentliche Gestaltungsmerkmal dar. Er ist ein abgeschlossenes Vorgangselement mit Fachinhalten und ist einem verantwortlichen Bearbeiter oder einer Gruppe von Bearbeitern (Rolle) zugeordnet. Zumeist erfolgt mit dem Abschluss eines Vorgangsschrittes auch die Übergabe des Vorgangs an einen anderen Bearbeiter bzw. eine andere Rolle. Ein Vorgangsschritt wird durch Erreichen eines Ereigniszustandes (durch ein Ereignis oder eine Kombination von Ereignissen) aktiviert. Es gibt unterschiedliche Ereignistypen, die auf vielfältige Weise ausgelöst werden können. Die Schrittfolge wird so durch das Eintreten von Ereignissen und entsprechende Aktivierungsbedingungen (Ereigniszustände) für Vorgangsschritte festgelegt.

Aktivitäten sind Hilfsmittel zur Beschreibung und weiteren organisatorischen Unterteilung eines Vorgangsschrittes. Sie können in verschiedenen Vorgangsschritten gleichartig vorhanden sein und müssen zunächst keiner bestimmten Ablaufsequenz folgen. Gleichwohl kann ihre Ausführung in bestimmten Schritten für den Sachbearbeiter Pflicht sein, oder die Ausführbarkeit von Aktivitäten kann an den Bearbeitungs- und Datenzustand des Vorgangs gekoppelt sein. Die Aktivität ist die kleinste Einheit, zu der *EasyFlow* Ablaufinformationen protokolliert (Wer hat wann diese Aktivität wie lange mit welchem Ergebnis bearbeitet?). Es besteht zudem die Möglichkeit, in Aktivitäten Fachdaten abzulegen und so den vollständigen Geschäftsvorfall in der Prozesshistorie zu archivieren. Unterschiedliche, vom Kunden erweiterbare Aktivitätstypen ermöglichen das gezielte Ansprechen von Anwendungskomponenten.

*EasyFlow*-Subprozesse sind eigene Vorgänge, die aus einem Prozess heraus aufgerufen werden können. Je nach Definition wartet der Vaterprozess, bis der Subprozess abgeschlossen ist. Dem Subprozess werden beim Aufruf bestimmte Werte des Vaterprozesses vererbt.

Die Vorgangsdefinition erfolgt in der Regel top-down von der Vorgangsklasse über Vorgänge und Vorgangsschritte bis hin zur Schrittfolge und den Aktivitäten innerhalb der Schritte. Mittels der Vorgangsklasse wird so ein Metavorgang definiert, dessen Schritte, Ereignisse und Aktivitäten auf die abgeleiteten Vorgänge übertragen werden. Innerhalb der abgeleiteten Vorgänge können Aktivitäten und Ereignisse verändert sowie die Schrittfolge variiert werden.

Die Definition eines Vorgangs mittels Vorgangsklasse, Vorgangsschritten, Ereignissen und Aktivitäten stellt das Regelwerk dar, gemäß dem jeder einzelne, spezifische Fall abzulaufen hat. Für jeden definierten Vorgang wird es demnach zu jedem Zeitpunkt eine unbestimmte Anzahl von Fällen geben, die nach den in ihnen definierten Regeln ablaufen und sich an den unterschiedlichsten Stellen innerhalb des Ablaufs befinden. Diese Fälle sind Instanziierungen der jeweiligen Vorgänge.

In *EasyFlow* wird strikt zwischen der Ablaufsteuerung des Vorgangs (was ist wann zu tun) und der Bearbeiter-Ermittlung (wer darf/muss es tun) getrennt. Bearbeiter eines Schrittes werden erst dann ermittelt, wenn der Schritt durch die Ablaufsteuerung (Ereignisse) aktiviert wurde. Für die Ermittlung der Adressaten eines Schrittes sind Rollen, Aufträge und Auftragsbeziehungen<sup>16</sup> sowie Systemadressaten relevant. Akteure im Workflow können sowohl menschliche Sachbearbeiter als auch Systeme wie Prozessautomaten, Backendsysteme etc. sein.

Weitere Merkmale von *EasyFlow*:

- automatische ereignisbasierte Ablaufsteuerung
- flexible Fristenverwaltung mit Eskalation
- Checklisten-Funktionalität
- Unterstützung von marktüblichen Entwicklungsumgebungen und Standards
- modellierbares Vier-Augen-Prinzip
- dynamisches Adressierungsverfahren für Schritte
- umfassende Vertretungsregelung
- verzögerungsfreie Weiterleitung von Schritten
- Darstellung der Vorgangshistorie durch standardisierte Dialoge
- Wiedervorlage auf Schrittebene
- Gruppentätigkeitsliste

---

<sup>16</sup> Die Auftragsbeziehungen in *EasyFlow* bieten die Möglichkeit, Schrittheadressierungen in Abhängigkeit von Prozessvariablen (z.B. Kredithöhe, Postleitzahl, etc.) dynamisch zur Laufzeit durchzuführen.

## Software-Architektur

Basierend auf einer Datenbank werden die Datenbankzugriffe über eine gekapselte allgemeine funktionale Schicht und eine Zugriffsschicht realisiert. Diese können je nach Anforderung der Anwendung mit SQL Windows oder C++ verwendet werden. In der Zugriffsschicht ist nicht nur die Workflow-Funktionalität von den Fachanwendungen wie Textverarbeitung oder ERP gekapselt, sondern auch die Möglichkeit geschaffen worden, weitere Datenbankprodukte zu verwenden. Dies wurde durch die Verwendung der Rogue-Wave-Klassenbibliotheken ([www.roguewave.com](http://www.roguewave.com)) ermöglicht. Oberhalb der Zugriffsschicht sind die Funktionen unterteilt in das Modul Vorgangssteuerung und die Module, die in einem Geschäftsprozess verwendet werden. Dies kann eine Textverarbeitung, eine Hostanbindung, eine E-Mailsystem-Anbindung oder auch ein ERP-System sein. In dem Modul Vorgangssteuerung sind alle Funktionen zur Steuerung und Verwaltung der Prozesse implementiert. Diese werden durch bereits realisierte *EasyFlow*-Dialoge verwendet und können auf funktionaler Ebene auch von eigenen Anwendungen verwendet werden.

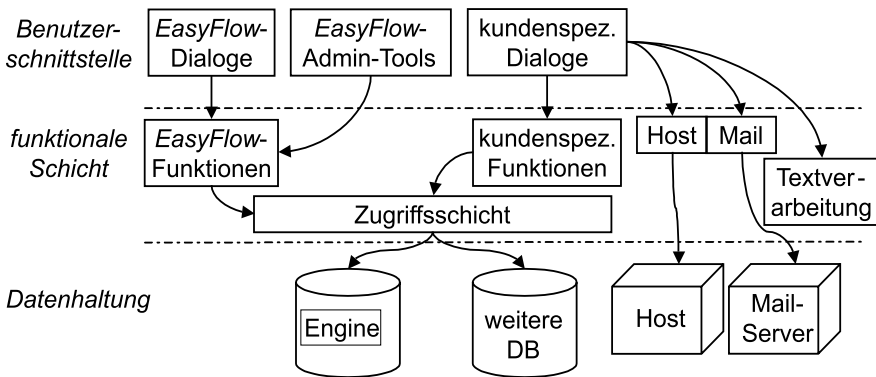


Abbildung 8.10. *EasyFlow*-Architektur

Im Produktumfang ist ein Standard-Client (vgl. Abbildung 8.11) enthalten, der bereits grundlegende Dialoge wie Suchmaske, Tätigkeitsliste und Prozesshistorie implementiert hat. Die Workflow-Engine von *EasyFlow* kann auf drei verschiedenen Ebenen angesprochen werden. Auf der untersten Ebene können direkt die in der Datenbank implementierten Funktionen über SQL aufgerufen und für die Prozesssteuerung verwendet sowie auf Serverebene an Drittanbieter angebunden werden. Die nächste Ebene ist die Verwendung der Zugriffsschicht-APIs (APL und DLL), die als Kapselung und Ergänzung zu den in der Datenbank implementierten Funktionen zu sehen sind. Die oberste Ebene stellen die Standarddialoge der Vorgangssteuerung dar.

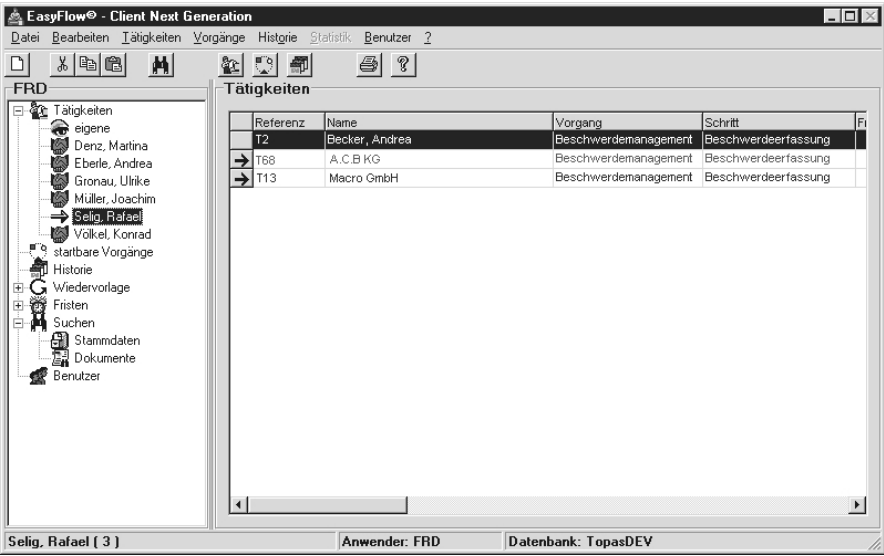


Abbildung 8.11. EasyFlow-Standard-Client

In der folgenden Aufstellung wird die implementierte Funktionalität in den drei Ebenen dargestellt.

Tabelle 8.2. Implementierte Funktionen

Ebene	Funktion
Datenbank	Alle vorgangs- und ablaufspezifischen Funktionen: Definition der Prozesse, Steuerung der Prozesse und Historie der Prozesse.
API	Zusätzlich zu den oben aufgeführten Funktionen sind implementiert: Nachverfolgung (Tracing) der Funktionsaufrufe auf Client-Seite, Fehlerverlaufsaufzeichnung , Host-Connectivity, Unterstützung von Mehrsprachigkeit, Kapselung aller Texte in der Datenbank.
Präsentation	Zusätzlich zu den für die oben genannten Ebenen aufgeführten Funktionalitäten existieren Dialoge wie: Suchmaske, Tätigkeitsliste, Standardschrittfenster mit Aktivitätenliste, Prozesshistorie und Fehlerdialog

Systemanforderungen

Client:

- ab Pentium III, kompatibles System
- ab 64 MB RAM (je nach Betriebssystem)

- 20 MB Festplattenplatz
- MS-Windows 98, ME, NT, MS-Windows 2000. XP

Datenbanken-Server:

- Oracle ab Version 7 und alle Datenbankprodukte, die interne Funktionen unterstützen
- weitere Angaben abhängig von Datenbankprodukt

Netzwerk:

- TCP/IP
- DB-Netzwerkverbindung

Die Vor- und Nachteile sind nicht nach den oben genannten Kategorien sortiert, sondern in der Reihenfolge ihrer Bedeutung.

Vorteile von *EasyFlow*

- Durch die 100%ige Integration der Funktionalität können alle Features des Datenbankherstellers wie Performance-Optimierung, Skalierbarkeit, Verfügbarkeit und Sicherung mit geringem Aufwand verwendet werden.
- geringe Systemanforderungen bzgl. Festplattenplatz und Prozessorleistung von Servern und Clients
- Standardfunktionalitäten, wie ein Vier-Augen-Prinzip, dynamische Adressierungs- und Genehmigungsverfahren, Vertretungsregelungen und die automatische Wiedervorlage, Fristüberwachung von relativen (z.B. in 10 Tagen) und absoluten Fristen (z.B. am 31.12.)
- hoher Freiheitsgrad für die Anbindung der Vorgangssteuerung (Datenbankebene, mit Dialogen von Drittanbietern und durch *EasyFlow*-Standarddialoge)
- Unterstützung einer Entwicklungs-, Test- und Produktionsumgebung
- geringe Lizenzkosten
- Auslieferung des Sourcecodes möglich

Nachteile von *EasyFlow*

- Grafische Modellierung der Prozesse nicht durch einen eigenen grafischen Editor möglich.
- Administrations- und Definitionswerkzeug sind aktuell nur in SQL Windows realisiert und müssen deshalb mit der SQL-Windows-Runtime-Umgebung betrieben werden.
- Relativ „mächtiger“ Client, da die Funktionalität nicht auf dem Server ausgeführt wird, sondern auf dem Client. Dies trifft aber nur zu, wenn ein Client mit *EasyFlow*-Client-Standardfunktionalität realisiert wird. Bei der Verwendung der reinen Vorgangssteuerung ist dies nicht relevant.
- Die Architektur der *EasyFlow*-Standardkomponenten (datenbankbasierte Workflow-Engine, Zugriffsschicht und Workflow-Client) entspricht nicht den heuti-

gen modernen Architekturvorstellungen. Dies kann jedoch kompensiert werden, wenn die Workflow-Engine in direkter Verbindung mit einer serverbasierten Anwendung, welche durch eine J2EE-Architektur realisiert wurde, genutzt wird.

## **8.2 Auswahlkriterien für ein WfMS**

Die aufgeführten Workflow-Produkte sollen nur einen Einblick in die unterschiedlichen Technologien geben. Für die Anforderungen eines Kunden oder dessen Unternehmens ist eine Bedarfsanalyse notwendig, in der die Anforderungen des Unternehmens analysiert, gewichtet und bewertet werden. Darauf folgend ist eine Evaluierung der Workflow-Produkte durchzuführen, in der die Auswahlkriterien an den Anforderungen ausgerichtet sind. Der folgende Abschnitt gibt einen Überblick über mögliche Auswahlkriterien, um eine solche Evaluierung zu erleichtern.

Es wird direkt mit den Produkteigenschaften begonnen, da eine kommerzielle und wirtschaftliche Betrachtung des Produktherstellers sich nicht von anderen Produktevaluierungen unterscheidet – eine detaillierte Betrachtung der Firmengröße, der Anzahl der verkauften Lizenzen, der Firmeneigentümer usw. wird nicht berücksichtigt.

Die Erfahrung hat gezeigt, dass eine Entscheidungsmatrix für die Durchführung einer Produktevaluierung eine gute Basis ist, um anschließend eine Entscheidung zu treffen oder eine bereits getroffene Entscheidung eventuell zu bekräftigen. Der Aufbau einer solchen Matrix kann über den Link [www.workflow-based-integration.org](http://www.workflow-based-integration.org) in Form einer Excel-Datei geladen werden. In einer solchen Matrix sind die zuvor analysierten Anforderungen an das Produkt gruppiert und durch die Beteiligten gewichtet worden. Diese Gewichtung findet in zwei Stufen statt. Zum einen werden die einzelnen Anforderungen innerhalb der Gruppe gewichtet, um so die Bedeutung des jeweiligen Kriteriums hervorzuheben. Für den Stellenwert verteilt man Punkte, am Besten ähnlich der Formel 1, d.h. für das Kriterium von hoher Bedeutung vergibt man den Platz 1 mit zehn Punkten und ein unbedeutendes Kriterium erhält Platz 8 mit einem Punkt. Es sollte keine Gleichverteilung zwischen den Plätzen 1 und 8 erfolgen, sondern beispielsweise eine Verteilung 10, 8, 6, 5, 4, 3, 2 und 1 vorgenommen werden. Die zweite Stufe der Gewichtung ist die Gewichtung auf Gruppenebene, d.h. welche Bedeutung misst man den Gruppen untereinander zu. In Abbildung 8.12 ist ein Auszug aus einer solchen Entscheidungsmatrix dargestellt.



Entscheidungsmatrix				Produkt 1		Produkt2		Produkt3	
Kriterien	Beschreibung	Gewichtung		Bewertung	Zw-Erg.	Bewertung	Zw-Erg.	Bewertung	Zw-Erg.
Kriteriumgruppe A			1 <b>D</b>		0 <b>E</b>		0		0
Kriterium A.1					0		0		0
Kriterium A.2					0		0		0
Kriterium A.3		<b>A</b>		<b>B</b>	0 <b>C</b>		0		0
Kriterium A.4					0		0		0
Kriteriumgruppe B			2		0		0		0
Kriterium B.1					0		0		0
Kriterium B.2					0		0		0
Kriterium B.3					0		0		0
...			...		...		...		...
Kriteriumgruppe E			1		0		0		0
Kriterium E.1					0		0		0
Kriterium E.2					0		0		0
Kriterium E.3					0		0		0
			Summe		0 <b>F</b>		0		0

Abbildung 8.12. Auszug aus einer Entscheidungsmatrix

Unter A wird die Gewichtung des Kriteriums innerhalb einer Gruppe eingetragen, während unter D die Gewichtung der Kriteriengruppen auf Gruppenebene durchgeführt wird. Sind beide Arten der Gewichtung durchgeführt, kann mit der Bewertung der einzelnen Produkte begonnen werden. Durch Eintragen von B kann das Produkt aus A und B gebildet werden. Dieses Produkt wird in C eingetragen, dann innerhalb einer Gruppe aufsummiert und mit der Gewichtung D multipliziert. Das Ergebnis wird in E eingetragen. Diese Vorgehensweise wird für alle Kriterien durchgeführt und die Ergebnisse aller Kriteriengruppen aufsummiert (F).

Die Einteilung der Kriteriengruppen kann neben den Daten über den Produkthersteller wie folgt aussehen:

- Modellierung
- Prozessunterstützung
- Technologie allgemein
- Technologie Server
- Technologie Client
- Schnittstellen
- Werkzeuge

Die Kriteriengruppen werden in den folgenden Tabellen aufgeführt.

---

**Modellierung**


---

Kriterium	Beschreibung
Workflow-Typen	Welche Workflow-Typen werden durch das Produkt unterstützt (Produktion, Ad-hoc, dokumentenorientiert oder Mischformen)?
Fristenverwaltung	Welche Arten (relativ/absolut) von Fristen können modelliert werden?
Unterstützung durch Wizards	Werden Wizards für das Arbeiten mit dem WfMS angeboten?
Referenzmodell	Orientiert sich der Produkthersteller an der WfMC, um mögliche Anbindungen effektiv durchführen zu können?
Unterstützung von Prozess-templates	Werden durch den Produkthersteller Prozesstemplates (Vorlagen) angeboten oder können diese durch den Anwender erstellt werden? Die Vorlagen können als unternehmensweite Basis für die Modellierung von Prozessen verwendet werden.
Unterstützung von Prozessdokumentation	Kann aus dem modellierten Prozess eine Prozessdokumentation generiert werden?
Unterstützung von Prozessstandards	Werden Prozessstandards wie z.B. BPEL oder BPML unterstützt?
Unterstützung von UML	Unterstützt das grafische Modellierungs-Werkzeug die UML für die Modellierung von Prozessen?

---

**Prozessunterstützung**


---

Kriterium	Beschreibung
Prozesshistorie	In welcher Form werden die Prozessdaten historisiert und wie können diese wieder dargestellt werden?
Versionskontrolle von Prozessen	Werden Versionskontrollen unterstützt, und wenn ja, in welcher Form? Können Prozesse zu einem bestimmten Stichtag die Version wechseln? Wie werden die bereits gestarteten Prozesse in der alten Version zu Ende geführt?

---

**Technologie (allgemein)**


---

Kriterium	Beschreibung
Verständlichkeit	Ist das Produkt durch seine Werkzeuge und seine Architektur leicht verständlich, um die Aufwände für die Realisierung und Wartung gering zu halten?
Technologische Architektur	Auf welcher Architektur basiert das WfMS (z.B. J2EE, Microsoft Architecture)?
Anbindung von Applikations- und Webservern	Ist in der Architektur die Anbindung von Applikations- und/oder Webservern vorgesehen?

**Technologie (allgemein)**

Kriterium	Beschreibung
Protokollieren aller Prozesslaufzeiten	Werden alle Prozesslaufzeiten (Liegezeit, Bearbeitungszeit, Transportzeit) pro Geschäftsprozessinstanz aufgezeichnet?
Vorhandene Produkte	Kann das WfMS in die bestehende IT-Landschaft eingebunden werden oder kann aufgrund der verwendeten unterschiedlichen Technologien eine Anbindung oder auch eine Integration nicht oder nur mit hohem Aufwand gewährleistet werden?
Kommunikationstechniken	Welche Technologien und Protokolle liegen der Kommunikation zwischen den Modulen des WfMS zu Grunde?
Protokollieren aller Prozessdaten zur Erstellung einer Kostenkontrolle	Werden alle notwendigen Daten protokolliert, um eine Kostenanalyse durchführen zu können?
Betriebssystemabhängigkeit	Auf welchen Betriebssystemen kann das WfMS betrieben werden?
Sicherheit der Prozesse und der Prozessdaten (Backup, Recovery, Restart)	Welche Verfahren werden unterstützt, um eine hohe Ausfallsicherheit und Datensicherheit zu erreichen?
J2EE-Architektur	Ist der J2EE-Standard eingehalten oder können über bestehende Schnittstellen J2EE-Lösungen angebunden werden?
Datenbanksysteme	Welche Datenbanken (Datenbank-Produkte) können angebunden werden?
Datenbankverbindung	Über welche Technologie sind die Datenbanken angebunden (JDBC, ODBC, datenbankspezifisches Protokoll)?
Adaptorentechnologie	Können Fremdsysteme über Adapter angebunden werden? Auf welcher Technologie basiert die Anbindung?
Unterstützung von Mehrsprachigkeit	Können mehrere Sprachen in den Dialogen unterstützt werden?
API u. welche Technologien	Welche Möglichkeiten werden für die Anbindung und die Integration des WfMS angeboten?
Web-Service-Unterstützung	Besteht die Möglichkeit, Web Services anzubinden und Teile des WfMS als Web Service anzubieten?
Unterschiedliche Datentypen für Prozessdaten	Können den Prozessdaten unterschiedliche Datentypen zugewiesen werden?
Dynamische Dialoge	Können aufgrund von Prozessdaten Dialoge während der Laufzeit verändert werden?
Sourcecode des WfMS	Kann der Kunde den Sourcecode des WfMS ausgeliefert bekommen? Dies erhöht die Investitionssicherheit, da man nicht von der wirtschaftlichen Situation des Produktherstellers abhängig ist.
Ablage der Prozessdaten	In welcher Form werden die Prozessdaten im System gespeichert?

---

**Technologie (Server)**


---

Kriterium	Beschreibung
Unterstützung von Directories (LDAP, X.500)	Können Directories für das Rechte/Rollen-Modell angebunden bzw. ausgelesen werden?
Multiserver/Clusterfähig	Kann das WfMS auf mehrere Server oder Cluster verteilt werden, um eine bessere Skalierung des Systems zu erhalten?

---



---

**Technologie (Client)**


---

Kriterium	Beschreibung
Standard-Client	Wird ein Standard-Client mitgeliefert?
Web-basierter Standard-Client	Wird ein web-basierter Standard-Client mitgeliefert?
Client-Arten	Welche Client-Arten sind möglich (Java-Client, Web-Client, ...)?
Unterstützung von Client-Personalisierung	Können die Anwender die Ansichten ihres Clients ihren Wünschen anpassen (Customizing)?
Schnittstellen	
Access und Authentication	Werden durch das WfMS die drei A's unterstützt (Autorisierung, Authentifizierung und Auditing)?
Schnittstellen zu Modellierungswerkzeugen	Welche Modellierungswerkzeuge können angebunden werden und anhand von welchen Formaten (z.B. XML) wird dies realisiert?

---



---

**Werkzeuge**


---

Kriterium	Beschreibung
Grafisches Formular-Werkzeug	Können die Dialoge mit grafischer Unterstützung erstellt werden?
Unterstützung von Organisationen	Können in dem Modellierungswerkzeug des Produktherstellers Organisationen definiert werden?
Grafisches Modellierungswerkzeug	Wird durch den Produkthersteller ein grafisches Modellierungswerkzeug angeboten?
Simulations-Werkzeug	Beinhaltet das Modellierungswerkzeug des Produktes eine Simulationskomponente, mit der die modellierten Prozesse getestet werden können, oder können Lösungen von Drittanbietern (ARIS, Adonis) angebunden werden?
Administrationswerkzeug	Welchen Umfang hat das Administrationswerkzeug?
Analyse-Werkzeug	Werden Werkzeuge ausgeliefert, mit denen eine Analyse der laufenden und beendeten Prozesse durchgeführt werden kann, oder können Lösungen von Drittanbietern angebunden werden?
Reporting-Möglichkeiten	Welche Reporting-Möglichkeiten werden angeboten und welche Formate werden unterstützt?

---

### 8.3 Eigenentwicklung eines WfMS

Das Ergebnis einer Evaluierung kann sein, dass keines der betrachteten WfMS-Produkte für ein Unternehmen in Frage kommt. Dies tritt dann ein, wenn keines der am Markt befindlichen Produkte sich in die bestehende IT-Architektur integrieren lässt oder der Aufwand der Integration nahezu einer Eigenentwicklung entspricht. Ist dies der Fall, so ist zu untersuchen, wie hoch der Entwicklungsaufwand für eine Eigenentwicklung mit den benötigten Prozesssteuerungsfunktionalitäten ist, und welche Produkte in das zu erstellende WfMS integriert werden sollen. Als Beispiel kann an dieser Stelle ein Modellierungswerkzeug genannt werden, welches durch eine Export-Schnittstelle die modellierten Prozesse in einem Dateiformat zur Verfügung stellen kann. Hat man sich für CORBA als eine technologische Basis entschieden, kann die Wahl für die Realisierung einer Workflow-Engine auf einen Produkthersteller eines ORBs fallen. Eine mögliche „Zwitterlösung“ besteht darin, den Kern eines WfMS zu kaufen und diesen dann weiterzuentwickeln. Um die Entscheidung für oder gegen eine Eigenentwicklung treffen zu können, ist die Berücksichtigung der nachfolgend aufgeführten Kriterien hilfreich.

Kriterium:	Aufwand/ Kosten
Eigenentwicklung:	Die Kosten orientieren sich daran, ob für die Entwicklung externes Know-how eingekauft werden muss, welche Module durch Standardprodukte eingekauft werden können und wie groß der Funktionsumfang der benötigten Module sein soll.
Produkt:	Es fallen neben den Lizenzkosten für die einzelnen Module wie Workflow-Engine, Reporting und Modellierung auch Kosten für einen Wartungsvertrag bzw. für Updates des Produktes an.
Anmerkung:	Um die Kosten gegenüberstellen zu können, sollte die Kostenschätzung einen Zeitraum von mindestens drei bis fünf Jahren berücksichtigen.

Kriterium:	Funktionsumfang
Eigenentwicklung:	Der Umfang der Funktionen kann selber bestimmt werden, und auch die Reihenfolge der Realisierung der Funktionen kann nach eigenen Prioritäten vorgenommen werden.
Produkt:	Da die Realisierung eines Produktes darauf ausgelegt ist, möglichst standardisierte Funktionen anbieten zu können, liegen zum Teil Funktionen vor, die gar nicht oder nur mit zusätzlichem Aufwand an die unternehmensspezifischen Prozessanforderungen angepasst werden können.
Anmerkung:	Der Vorteil einer Eigenentwicklung liegt darin, dass diese mit Beschränkung auf den benötigten Funktionsumfang realisiert werden kann und sich die Lösung deshalb „schlanker“ gestaltet.

Kriterium:	Erweiterung des WfMS
Eigenentwicklung:	Durch eine Phasenplanung können Erweiterungen kurz-, mittel- und langfristig geplant und den aktuellen Anforderungen angepasst werden.
Produkt:	Bestimmte Erweiterungen bzw. Anpassungen sind nur durch den Hersteller möglich, da nicht alle Funktionen für den Kunden dokumentiert bzw. zugänglich sind.
Anmerkung:	Bei einer Eigenentwicklung ist es wichtig, eine mit einer Produktversionisierung zu vergleichende Versionsplanung einzuführen, um die Verfügbarkeit einer neuen Funktionalität frühzeitig zu kennen.
Kriterium:	Integration
Eigenentwicklung:	Die Integration in die bestehende IT-Architektur wird nur durch eventuell eingeplante Frameworks oder hinzugekaufte Module eingeschränkt.
Produkt:	Es besteht die Gefahr, dass sich das WfMS nicht in der Version oder in späteren Versionen in die bestehende IT-Architektur integrieren lässt.
Anmerkung:	Es ist wichtig, mit dem ausgewählten Produkthersteller über dessen Planung der zukünftigen Produktarchitektur zu sprechen.
Kriterium:	Projektdurchführung
Eigenentwicklung:	Wird die Prozesssteuerung und die fachliche Lösung der Prozesse parallel durchgeführt, steht erst zu einem relativ späten Zeitpunkt im Projekt eine prototypische Lösung zur Verfügung.
Produkt:	Aufgrund des vorhandenen Produktes ist eine Basis vorhanden, mit der eine prototypische Lösung schnell realisiert werden kann und im Gesamtprojektverlauf die meisten Aufwände für die Realisierung der fachlichen Anwendung anfallen.
Anmerkung:	Bei einer Eigenentwicklung ist eine Trennung zwischen Prozesssteuerung und fachlicher Lösung erstrebenswert, da so die Projektorganisation ähnlich einer Implementierung eines Produktes durchgeführt werden kann und das Projekt so in überschaubare Teilprojekte aufgeteilt werden kann. Bei einer Vermischung von WfMS- und Fachfunktionalität besteht zudem die Gefahr, dass im Laufe der Zeit eine nicht mehr handhabbare monolithische Lösung entsteht.

Kriterium:	Risiko
Eigenentwicklung:	Das Risiko für eine erfolglose Entwicklung ist dann sehr hoch, wenn im Vorfeld keine detaillierte Planung für die Eigenentwicklung durchgeführt und die Versionsplanung nicht auf Produktniveau organisiert wird.
Produkt:	Das Risiko ist für spezielle Lösungen gegenüber der Eigenentwicklung etwas geringer bzw. verlagert sich, da zumindest die Produkterfahrung durch den Hersteller eingekauft werden kann. Durch Teststellungen und Referenzinstallationen kann das Risiko weiter verringert werden. Andererseits liegt ein nicht unerhebliches Risiko in der wirtschaftlichen Entwicklung des Herstellers.
Anmerkung:	Risiken sind in beiden Fällen mit lediglich anderen Ausrichtungen vorhanden. Falls der Produkthersteller keine ausreichende Kundenbasis aufweisen kann, kann dies für den mittelfristigen Projektverlauf aufgrund von nicht vorhandenem Sourcecode des Produktkerns sehr kritisch werden. Dies kann man durch entsprechende vertragliche Regelungen minimieren, beispielsweise durch die Aushändigung des Sourcecodes im Falle einer Insolvenz (dies behebt allerdings nicht das Problem des fehlenden fachlichen Know-hows zur weiteren Pflege des Produktes).
Kriterium:	Zukunftssicherheit
Eigenentwicklung:	Die Zukunft der Eigenentwicklung hängt von dem IT-Budget des Unternehmens ab.
Produkt:	Es wird so gut wie kein Einblick in die zukünftige Produktausrichtung gewährt und i.d.R. bestimmt der Produkthersteller die Ausrichtung des Produktes.
Anmerkung:	Ist analog mit dem Kriterium Risiko zu bewerten.

Die aufgezählten Kriterien lassen darauf schließen, dass auch hier ein ausgeglichener Mix aus Eigenentwicklung und Produkt eine gute praktikable Lösung darstellt. Als Beispiel sei hier das Produkt *EasyFlow* genannt, welches mit den Sourcen gekauft werden kann, so dass auf dieser Basis die Realisierung einer Eigenentwicklung möglich ist. Eine weitere Option ist die Integration von Open-Source-Lösungen in eine Eigenentwicklung. In diesem Fall können (mit gewissen Einschränkungen je nach Open-Source-Lizenz und Hersteller) Weiterentwicklungen des WfMS genutzt werden, und man kann sich auf die Realisierung der fachlichen Lösung konzentrieren. Da die Weiterentwicklung der Open-Source-Lösung in der verteilten „Community“ liegt, kann davon ausgegangen werden, dass die Pflege z.B. einer Workflow-Engine grundsätzlich gesichert ist.

## 8.4 Modellierungswerkzeuge

An dieser Stelle wird auf eine detaillierte Beschreibung von Modellierungswerkzeugen verzichtet, da der Fokus des Buches auf den WfMS liegt und diese über XML-Schnittstellen Modelle aus unterschiedlichen Modellierungswerkzeugen importieren können oder auch grafische Editoren zur Verfügung stellen, um Modelle direkt für das WfMS zu erstellen. Die kurze Darstellung von zwei am Markt erhältlichen Modellierungswerkzeugen soll an dieser Stelle ausreichen.

### 8.4.1 ADONIS

ADONIS wurde von der Firma BOC GmbH entwickelt und kann für die Erhebung, Modellierung, Analyse und Simulation von Geschäftsmodellen verwendet werden. Anschließend können die erstellten Modelle in verschiedene Zielsysteme (WfMC, CASE-Tools, Dokumentationen etc.) überführt werden. ADONIS ist ein Client/Server-Mehrbenutzersystem, das objektorientiert aufgebaut ist und ein umfangreiches Customizing ermöglicht.

Mit dem *Geschäftsprozessmanagement-Toolkit* können mit Hilfe der grafischen Elemente die Modelle erfasst, analysiert, simuliert und evaluiert werden. Das Toolkit besteht aus verschiedenen Komponenten, welche die folgenden Vorgehensschritte unterstützen:

Die *Erhebungskomponente* unterstützt die Erfassung von Daten, welche für die Modellierung von Geschäftsprozessen wichtig sind. Aufgrund dieser Daten werden die Modelle mit der *Modellierungskomponente* erstellt. ADONIS bietet auch die Möglichkeit, die Modelle über eine tabellarische Eingabe zu erfassen.

Mit der *Analysekomponente* werden Abfragen auf die Modelle durchgeführt. Die rechnerische Auswertung von Prozessmodellen ermöglicht eine analytische Bewertung des Ablaufes.

Die *Simulationskomponente* ermöglicht die Simulation von Geschäftsprozessen. Es werden dafür vier Simulationsalgorithmen zur Verfügung gestellt, welche bei den Prozessmodellen eine Pfadanalyse, eine Belastungsanalyse, eine Auslastungsanalyse mit stationärer Betrachtung und eine Auslastungsanalyse mit nichtstationärer Betrachtung ermöglichen.

Die *Evaluationskomponente* bietet Mechanismen sowohl für die Evaluierung von Soll-Modellen als auch von produktiven Prozessen.

Mit der Import/Export-Komponente können die erstellten Modelle in ADL-Dateien exportiert werden bzw. umgekehrt ADL-Dateien in das *Geschäftsprozessmanagement-Toolkit* importiert werden (ADL = ADONIS Definition Language). Mit dem ADL-Format können die Modelle in andere ADONIS-Datenbanken transferiert werden, zusätzlich stellt die Datei eine Form der Datensicherung der



Modelle dar. Weiterhin können mit der Import/Export-Komponente die Modelle in FDL-Dateien für den Import in WfMS exportiert werden. Als Beispiel kann hier das WfMS FlowMark von IBM genannt werden.

Ein weiteres Toolkit ist das *Administrations-Toolkit*, welches Komponenten für die Verwaltung der Anwender und Benutzergruppen, der ADONIS-Bibliotheken und der Modelle enthält.

Mit Hilfe des *Dokumentations-Toolkits* können aus den ADONIS-Modellen RTF- und HTML-Dateien generiert werden. Die Modellinhalte können inklusive der graphischen Darstellung in Dokumente eingebunden bzw. über ein Intranet verteilt werden.

Als zusätzliches Werkzeug wird die *ADONIS-Prozesskostenanalyse* für die Kostensoptimierung angeboten. Dieses Werkzeug ermöglicht Auswertungen über Optimierungspotentiale. Dabei wird die Simulation des Prozessmodells genutzt, um Aussagen über die Prozesskosten treffen zu können.

#### 8.4.2 ARIS

ARIS bezeichnet ein von Prof. Scheer entwickeltes Konzept, welches in dem gleichnamigen Werkzeug realisiert wurde. Das Konzept basiert auf einer ganzheitlichen Betrachtung von Unternehmensprozessen und deckt neben der Modellierung auch die Optimierung von Prozessen ab. Das ARIS-Konzept, welches im Anhang beschrieben ist, wurde in diesem Tool realisiert. Hier basiert die Darstellung der Elemente auf EPKs, welche durch Prof. Scheer erweitert wurden (eEPK). Die IDS-Scheer AG bietet eine Vielzahl von zusätzlichen Produkten auf Basis des ARIS-Toolset an, die für

- Simulation und Auswertung,
- Prozesskostenrechnung,
- Qualitätsmanagement

sowie zur Unterstützung bei

- der Einführung und der Entwicklung von Balanced Scorecard auf der Basis von Geschäftsprozessen,
- Datawarehouse-Projekten,
- der Einführung von E-Commerce-Systemen und
- der Einführung von Wissensmanagement

eingesetzt werden können.

Neben dem ARIS-Toolset stehen dem Anwender Zusatzkomponenten zur Verfügung, die die oben genannten Funktionalitäten abdecken. Die bekanntesten werden Im Folgenden kurz beschrieben.

**ARIS Web Publisher**

Der Web Publisher unterstützt die web-basierte Einsicht in die Prozessmodelle. Es können so über das Internet Informationen über den Prozess und die Organisationsstrukturen dargestellt werden.

**ARIS Simulation**

Durch Simulation können anhand von Prozesskennzahlen die zuvor modellierten Prozesse simuliert werden. Dadurch können vor dem Produktiveinsatz der Prozesse Modellierungsfehler und Flaschenhälse erkannt und beseitigt werden. Es besteht auch die Möglichkeit, dass Messsonden in einem bestehenden Geschäftsprozess implementiert werden, um so praxisnahe Prozesskennzahlen zu erhalten.

**ARIS Easy Design**

Diese Komponente versetzt die Mitarbeiter der Fachabteilungen in die Lage, Prozesse zu gestalten und Reports sowie Dokumentationen der Prozesse zu erstellen. Sie enthält nicht alle Funktionen des konventionellen Modellierungs-Clients.

**ARIS Web Design**

Web Design unterstützt die web-basierte Modellierung von Prozessen und enthält weitgehend die Funktionen des konventionellen Modellierungs-Clients.

Bei beiden Modellierungswerkzeugen besteht die Möglichkeit, grafische Modelle zu erzeugen, die erstellten Modelle zu simulieren und diese aufgrund der Simulationsergebnisse zu optimieren. Durch die Import/Export-Schnittstelle können die erstellten Modelle in Workflow-Management-Systeme importiert werden, und die Prozesskennzahlen aus den WfMS können in das Modellierungswerkzeug reimportiert werden. Somit wird der Workflow-Regelkreis aus dem Kapitel Workflow unterstützt. Es ist aber darauf zu achten, dass es sich um unterschiedliche WfMS-Produkte handelt, d.h. bei der Evaluierung eines WfMS-Produktes muss auf die entsprechende Im/Export-Schnittstelle geachtet werden.

Weitere Modellierungswerkzeuge können aus der Studie „Business Process Management Tools: Eine evaluierende Marktstudie über aktuelle Werkzeuge“ des Fraunhofer Institut für Arbeitswirtschaft und Organisation IAO von 2002 entnommen werden.

**8.5 Kommentar**

Wie das Kapitel zeigt, ist nicht nur der Kern eines WfMS für eine erfolgreiche Realisierung einer prozessbasierten Integration relevant, sondern es sind neben den Modellierungswerkzeugen auch die Schnittstellen für die Anbindung von Fremdsystemen von erheblicher Bedeutung.

Die hier skizzierten Workflow-Management-Systeme sind, wie angekündigt, in unterschiedlichen Technologien realisiert. Somit kann man aufgrund der vorhan-

denen Basis im Unternehmen und den Anforderungen an die Lösung das am besten geeignete System auswählen. Die Produktmerkmale erlauben folgende Zusammenfassung:

CARNOT kann für eine J2EE-Lösung, also einer Inter-/Intranet-Lösung mit vielen unterschiedlichen Schnittstellen durch J2EE-Konnektoren, präferiert werden. Durch die Möglichkeit, Open-Source-Projekte (Datenbank- und Applikationsserver) mit in die Lösung zu integrieren, können auch die Lizenzkosten optimiert werden. Durch den umfassenden Lieferumfang an WfMS-Werkzeugen wie Modellierungs- und Administrator Desktop und den unterschiedlichen Client-Varianten (JAVA- und Web-Client) kann auf den Zukauf von Drittanbietern dieser Komponenten weitgehend verzichtet werden.

e-Work dagegen kann aufgrund der primär verwendeten Microsoft-Technologie für eine Windows-Umgebung im Unternehmen empfohlen werden. Ansonsten muss mindestens ein Microsoft-Server im Unternehmensnetz vorhanden sein. Durch die Möglichkeit, über einen Standard-Konnektor Anwendungen anbinden zu können, die mit den J2EE-Standard realisiert wurden, ist die WfMS-Lösung in der Lage, weitere Anwendungen über diesen Standard anzubinden. e-Work zeichnet sich besonders durch die einfache Modellierung eines Prozesses aus. Durch das mitgelieferte Modellierungswerkzeug kann effektiv ein Prozess erstellt und durch den Web-Client auch betrieben werden. Dadurch kann schnell ein Prototyp eines Prozesses erstellt und der Prozessablauf und verwendete Prozessdaten getestet werden.

Nach Meinung des Autors liegt der Unterschied der beiden Produkte nicht nur in den unterschiedlichen Technologien, sondern auch in der Offenheit, andere Produkte (Datenbank, Web- und Applikationsserver) einzubinden, die für den Betrieb des WfMS-Produktes benötigt werden. So kann durch die Architektur von CARNOT eine relativ große Auswahl von Fremdprodukten eingebunden werden, während bei e-Work für die Datenbanken nur Microsoft SQL-Server oder eine Oracle-Datenbank zur Verfügung stehen und als Web-Server lediglich der IIS (Internet Information Server) von Microsoft eingebunden werden kann.

Neben den Produkteigenschaften der WfMS-Produkthersteller sind auch die vorhandenen Partner ein nicht zu unterschätzendes Entscheidungskriterium. Es ist für jeden Produkthersteller notwendig, dass ausreichend Partner für die Realisierung von unterschiedlichen Schnittstellen zu weiteren Produkten vorhanden sind. So kann für ein mittelständisches Unternehmen die Schnittstelle zu einem ERP-System, welches nicht von SAP ist, von Interesse sein. Diese Schnittstellen z.B. zu Peoplesoft, Siebel oder Navision werden in der Regel durch Partner realisiert, die eine solche Anbindung oder auch Integration der Systeme auf unterschiedlicher technologischer Basis durchführen. Die beiden Produkte (CARNOT und e-Work) weisen eine SAP-Anbindung und eine ARIS-Anbindung auf bzw. diese befinden sich durch den Produkthersteller oder von Partnern in der Realisierung.

Die beiden oben genannten Produkte und *EasyFlow* unterscheiden sich in der verwendeten Technologie gravierend. Die *EasyFlow*-Engine ist zu 100-Prozent in einer Datenbank realisiert und kann somit vollständig im Hintergrund einer Datenbank-Anwendung oder einer Anwendung, in der eine Datenbank integriert ist, implementiert werden. Daraus ergibt sich der Vorteil, dass die Engine sich auf das Steuern der Prozesse konzentriert und somit sehr schlank realisiert wurde. Die implementierten Standardfunktionen heben sich zum Teil sehr stark von den Marktführern ab. So ist in *EasyFlow* bereits ein Vier-Augen-Prinzip und die Vertretungsregel integriert und kann ohne zusätzlichen Realisierungsaufwand verwendet werden. Auch die datumsgesteuerte Gültigkeit von Vorgangsversionen bringt Vorteile bei der Einführung neuer/geänderter Abläufe. Der größte Nachteil in Bezug auf effektive Modellierung liegt in den ausgelieferten Werkzeugen. Diese enthalten zwar alle notwendige Funktionen, um Prozesse zu modellieren und zu administrieren, aber sie weisen keine Möglichkeit der grafischen Darstellung der modellierten Prozesse auf. Die Prozessmodellierung wird durch ein Definitionswerkzeug, in dem die Prozesse modelliert werden. Dieser Nachteil kann jedoch durch die Verwendung des grafischen Modellierungswerkzeugs Adonis kompensiert werden, da eine Schnittstelle zwischen den beiden Produkten realisiert wurde.

Zu den drei vorgestellten WfMS-Produkten kann abschließend gesagt werden, dass alle drei ihre Stärken auf zum Teil sehr unterschiedlichen Gebieten haben und dadurch unterschiedliche kundenspezifische Anforderungen abdecken. Aber alle ermöglichen es, Geschäftsprozesse effektiv zu unterstützen bzw. zu automatisieren.

### **Kritische Anmerkung**

Die Auswahl eines WfMS kann auch durch eine Studie von beispielsweise Gartner oder GIGA stark beeinflusst werden. Bei deren Berücksichtigung liegt man nie völlig falsch, und wenn doch, kann man als Entscheider immer noch sagen, man hat ja den Marktführer/Studiensieger genommen, um dann die Schuld der Fehlentscheidung von sich weisen.

Hier zeigt sich aber auch eine Problematik bei der Entscheidungsfindung in Mitteleuropa: Ein solches Vorgehen ist nicht unbedingt zielführend für das Unternehmen und kann erhebliche wirtschaftliche Folgen haben. Denn die Studienergebnisse sind nicht so neutral, wie es häufig dargestellt wird: Die Berücksichtigung von Produkten in solchen Studien ist z.B. nicht ganz kostenfrei. Ist der Produkthersteller nicht bereit, die Kosten zu übernehmen, wird sein Produkt erst gar nicht in einer solchen Studie berücksichtigt. Damit ist aber eine objektive Beurteilung der am Markt vorhandenen Produkte nicht mehr möglich.

Der Aufwand für eine eigene Evaluierung ist nicht unerheblich, aber für eine zukunftssichere Entscheidung aus Sicht des Autors zwingend notwendig.

## 9 Template

Es werden Im Folgenden die Templates, die in dem Kapitel Workflow-Projekte aufgeführt wurden, beschrieben und dargestellt. Unter [www.workflow-based-integration.de](http://www.workflow-based-integration.de) stehen diese zum Download zur Verfügung.

### 9.1 Projektziele bestimmen

Ziel:

Gemeinschaftliches Verständnis über die Projektziele.

Vorgehen:

Die Ziele für das ganze Projekt sind festzulegen. Folgende Inputs dienen dazu, Projektziele zu definieren:

- vorhandene Benchmarks und Kundenbefragungen
- Anforderungen und Informationen aus dem Qualitätsmanagementsystem
- das Unternehmensumfeld (interne und externe Ressourcen, vorgegebener Termin- und Kostenrahmen, usw.)
- Ist das Ziel überprüfbar? Wenn ja, wie?

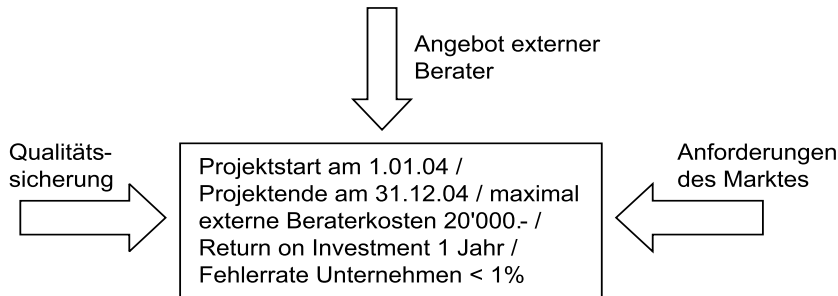
Spaltenbeschreibung:

Name des Ziels:	Unter welchem Namen soll das Ziel im Unternehmen genannt werden?
Beschreibung:	Kurze Beschreibung des Ziels
Strategie:	Gehört das Ziel zu der Unternehmensstrategie?
Überprüfbar:	Kann das Ziel quantitativ erfasst werden?
Kostenrahmen:	In welchem finanziellen Rahmen bewegt sich die Umsetzung bzw. Erreichung des Ziels?
Termin:	Welches Datum ist für die Umsetzung geplant?
Umfeld:	In welchem Umfeld befindet sich das Ziel?

Checkliste zum Vorgehen:

- Unterstützen die Projektziele die Unternehmensstrategie?
- Müssen Kundenbefragungen durchgeführt werden, die helfen die Projektziele zu identifizieren?
- Wer liefert den Input?
- Wer unterstützt wie zur Erreichung des Ziels?

Projektziel definieren (grafisches Beispiel):



Nutzen:

Die Ziele für das Projekt sind in Form von Messgrößen anzugeben. Die Ziele sollten, wenn möglich, für Zeit, Kosten und Qualität formuliert werden, z.B.: Projektstart am 1.01.04, Projektende am 31.12.04, maximal externe Beraterkosten 20.000.-, Return On Investment 1 Jahr, Fehlerrate Unternehmen < 1%, usw.

Unternehmens- und globale Projektziele		Projekt: Projektname					T-WbH-01																																			
<b>Tabellarisch:</b> <table border="1"> <thead> <tr> <th>Name des Ziels</th> <th>Beschreibung</th> <th>Strategie? (j/n)</th> <th>Überprüfbar? (j/n)</th> <th>Kostenrahmen (von/bis)</th> <th>Termin</th> <th>Umfeld</th> </tr> </thead> <tbody> <tr><td> </td><td> </td><td> </td><td> </td><td> </td><td> </td><td> </td></tr> <tr><td> </td><td> </td><td> </td><td> </td><td> </td><td> </td><td> </td></tr> <tr><td> </td><td> </td><td> </td><td> </td><td> </td><td> </td><td> </td></tr> <tr><td> </td><td> </td><td> </td><td> </td><td> </td><td> </td><td> </td></tr> </tbody> </table>								Name des Ziels	Beschreibung	Strategie? (j/n)	Überprüfbar? (j/n)	Kostenrahmen (von/bis)	Termin	Umfeld																												
Name des Ziels	Beschreibung	Strategie? (j/n)	Überprüfbar? (j/n)	Kostenrahmen (von/bis)	Termin	Umfeld																																				
<b>Grafisch:</b> <div style="height: 150px; border: 1px solid black;"></div>																																										
<div> <div></div> <div>Projektleiter: Projektleiter; Datum: 08.07.2004</div> <div></div> </div>																																										
<small>Dokumentversion: 0.1, Datum: 10.12.2003, Autor: J. Müller</small>																																										

Abbildung 9.1. Vorlage Projektziele bestimmen

## 9.2 Prozesslandkarte erstellen

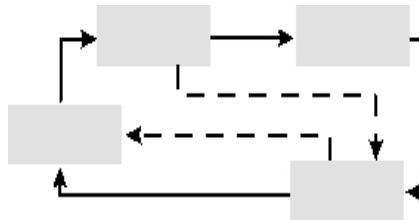
Ziel:

Überblick über die an den Unternehmenszielen beteiligten Geschäftsprozesse.

Vorgehen:

Jeder Geschäftsprozess kann als Black Box dargestellt werden. Die wichtigsten Datenflüsse werden mit gestrichelten (---) und die wichtigsten Materialflüsse mit durchgezogenen Linien dargestellt. Fließen Informationen oder Material in beide Richtungen, wird ein zweiseitiger ( $\leftrightarrow$ ) Pfeil gezeichnet.

Prozesse auswählen und darstellen:

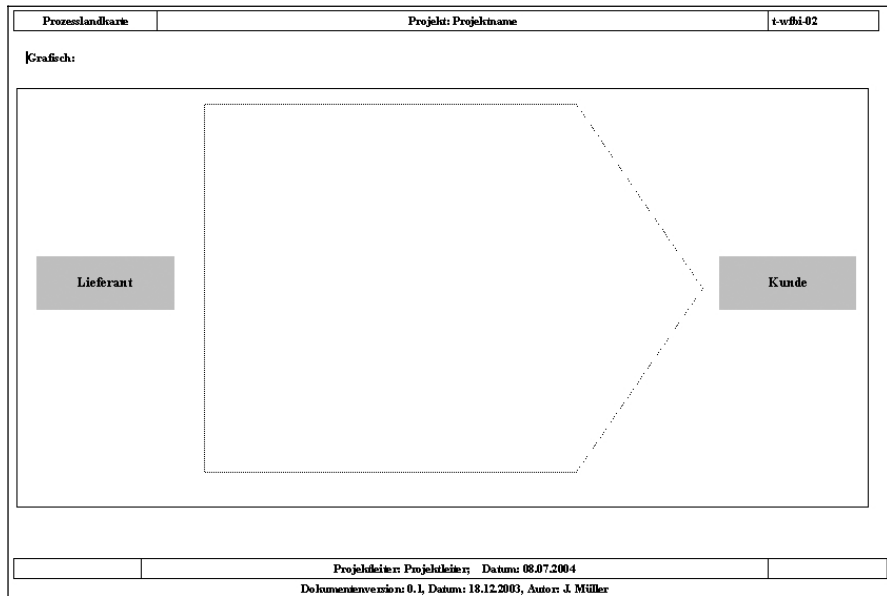


Checkliste zum Vorgehen:

- Sind alle In- und Outputs der entsprechenden Geschäftsprozesse identifiziert?
- Sind alle Funktionen oder Teilprozesse berücksichtigt?
- Sind alle internen und externen Kunden-Lieferanten-Beziehungen geklärt?
- Bietet die Prozesslandkarte eine Basis, um die darin beschriebenen Schnittstellen und damit zusammenhängende Probleme zu diskutieren?

Nutzen:

Gibt auf schnelle Weise einen Überblick, wie die Geschäftsprozesse zusammenhängen. Hilft dabei alle Geschäftsprozesse zu identifizieren. Die gefundenen Prozesse oder Funktionen repräsentieren den Istzustand. Alle für die aufgenommenen Prozesse oder Funktionen verantwortlichen Mitarbeiter werden in weiteren Schritten die enthaltenen Teilprozesse detaillieren.



**Abbildung 9.2.** Vorlage Prozesslandkarte erstellen

### 9.3 Erfolgsfaktoren ermitteln

**Ziel:**

Einheitliches Verständnis bezüglich der Unternehmensziele.

**Vorgehen:**

Zählen Sie die fünf wichtigsten Erfolgsfaktoren auf. Erfolgsfaktoren sind Ausprägungen der Firma, welche sich von anderen Firmen abheben und schwer imitierbar sind.

**Nutzen:**

Hilft, sich später auf die wichtigsten Prozesse konzentrieren zu können.



Erfolgsfaktoren ermitteln		Projekt: Projektname	t-wb1-03
Beschreibung des Geschäftes			
Nr.	Erfolgsfaktor (Name)	Beschreibung	
1			
2			
3			
4			
5			
Projektleiter: Projektleiter; Datum: 08.07.2004 Dokumentenversion: 0.1, Datum: 18.12.2003, Autor: J. Müller			

Abbildung 9.3. Vorlage Erfolgsfaktoren ermitteln

## 9.4 Prozesse bewerten

Ziel:

Geschäftsprozesse auswählen, die das Projektziel und die Erfolgsfaktoren am besten unterstützen.

Vorgehen:

Für jeden Geschäftsprozess (senkrecht aufgelistet) ist zu beurteilen, wie er die Erfolgsfaktoren beeinflusst. Ist der betrachtete Prozess sehr wichtig für den Erfolgsfaktor, so bekommt er drei Punkte. Kann ein Prozess den entsprechenden Erfolgsfaktor nicht beeinflussen, bekommt er einen Punkt.

Nun sind waagrecht die Summen zu bilden. Der Geschäftsprozess mit der höchsten Punktzahl hat die höchste Priorität. Sind aufgrund des Projektziels auch Verbesserungen aus dem Qualitätsplan durchzuführen, ist dies bei der Auswahl zu berücksichtigen.

Es werden nur so viele Prozesse ausgewählt, wie mit den vorhandenen Ressourcen optimiert werden können.



Vorgehen:

Alle Beteiligten der identifizierten Prozesse benennen und als Ansprechpartner dokumentieren.

Nutzen:

Diese Personen dienen als Vertreter der entsprechenden Abteilungen.

Projektbeteiligte ermitteln		Projekt: Projektname			t-wbi-05	
<b>Fachabteilungen</b>						
Nr.	Rolle	Rollenbeschreibung	Abteilungsbezeichnung	Rolle in der Abteilung	Name des Mitarbeiters	
1						
2						
3						
4						
...						
<b>Projektspezialisten</b>						
Nr.	Rolle	Rollenbeschreibung	Abteilungsbezeichnung	Rolle in der Abteilung	Name des Mitarbeiters	
1						
2						
3						
4						
...						
Projektleiter: Projektleiter; Datum: 06.07.2004 Dokumentversion: 0.1, Datum: 18.12.2003, Autor: J. Müller						

Abbildung 9.5. Vorlage Projektbeteiligte ermitteln

## 9.6 Prozesse bewerten und priorisieren

Ziel:

Identifizieren der wichtigsten Geschäftsprozesse.

Vorgehen:

- Aufnahme aller Geschäftsprozesse
- Aufführung der Zulieferprozesse
- Aufnahme aller Leistungen gegenüber dem Kunden (In-/Output)
- Bewertung der Messkriterien (Zeit, Kosten, Qualität)
- Prioritäten vergeben



## 9.7 Mengengerüst erfassen

### Ziel:

Erfassen des Mengengerüsts zur Beurteilung der Zulieferprozesse, um jene Prozesse mit den größten Aufwänden zu identifizieren.

### Spaltenbeschreibung:

Zulieferprozess:	Name des Zulieferprozesses
Verantwortlicher des Zulieferprozesses:	Name des Prozessverantwortlichen des Zulieferprozesses
Durchlaufzeit des Zulieferprozesses:	Enthält alle Zeiten vom Start bis zum Beenden des Zulieferprozesses.
Aktivitäten:	Name der Aktivitäten, die in diesem Prozess ausgeführt werden.
Aufwände:	Gemessene zeitliche Aufwände, die für die Aktivität notwendig sind.
Zeit pro Ausführung:	Zeitbedarf für jede Aktivität pro Prozessinstanz (Geschäftsvorfall)
Menge pro betrachtetem Zeitraum:	Anzahl der Ausführungen innerhalb des betrachteten Zeitraumes

### Nutzen:

Es wird der Ist-Zustand aufgenommen, um das Verbesserungspotential anhand des Mengengerüsts zu erkennen.



<b>Ressourcen erfassen</b>	<b>Projekt: Projektname</b>		<b>t-wfb1-08</b>																										
<table border="1"> <tr> <th rowspan="2">Geschäfts-/Zulieferprozess</th> <th colspan="2">Aufwand f. Personal p.a.</th> <th rowspan="2">Priorität</th> </tr> <tr> <th>Zeit (h)</th> <th>Kosten (€)</th> </tr> <tr><td> </td><td> </td><td> </td><td> </td></tr> <tr><td> </td><td> </td><td> </td><td> </td></tr> <tr><td> </td><td> </td><td> </td><td> </td></tr> <tr><td> </td><td> </td><td> </td><td> </td></tr> <tr><td> </td><td> </td><td> </td><td> </td></tr> </table>				Geschäfts-/Zulieferprozess	Aufwand f. Personal p.a.		Priorität	Zeit (h)	Kosten (€)																				
Geschäfts-/Zulieferprozess	Aufwand f. Personal p.a.		Priorität																										
	Zeit (h)	Kosten (€)																											
<table border="1"> <tr> <td> </td> <td><b>Projektleiter: Projektleiter; Datum: 08.07.2004</b></td> <td> </td> </tr> </table>					<b>Projektleiter: Projektleiter; Datum: 08.07.2004</b>																								
	<b>Projektleiter: Projektleiter; Datum: 08.07.2004</b>																												
<b>Dokumentversion: 0.1, Datum: 18.12.2003, Autor: J. Müller</b>																													

Abbildung 9.8. Vorlage Ressourcen erfassen

## 9.9 Verbesserungspotential bewerten

Ziel:

Identifizieren des Optimierungspotentials für die Geschäfts- und Zulieferprozesse.

Vorgehen:

- alle Geschäftsprozesse aufnehmen
- Zulieferprozesse aufführen
- alle Leistungen gegenüber dem Kunden aufnehmen (In/Output)
- Messkriterien bewerten (Zeit, Kosten, Qualität)
- Prioritäten vergeben





## 9.10 Teilprojekte identifizieren

Ziel:

Identifizieren der Teilprojekte mit den sich daraus ergebenden Maßnahmen.

Spaltenbeschreibung:

Teilprojekt:	Name, unter dem das Teilprojekt geführt wird.
Meilensteine:	Namen der einzelnen Meilensteine jedes Teilprojekts.
Maßnahmen:	Beschreibung der Maßnahmen, die zur Optimierung der Prozesse durchgeführt werden sollen.
Startdatum:	Datum, an dem die Maßnahmen beginnen sollen.
Zielfdatum:	Datum, an dem die Maßnahmen enden sollen.
Abhängigkeiten:	Abhängigkeiten zu anderen Meilensteinen und/oder Teilprojekten

Nutzen:

Dient als roter Faden für die Projektdurchführung und die Zielerreichung.

Teilprojekte identifizieren		Projekt: Projektname			twb1-10
Teilprojekt	Meilensteine	Maßnahmen	Startdatum	Zielfdatum	Abhängigkeiten

Projektleiter: Projektleiter; Datum: 08.07.2004	
Dokumentenversion: 0.1, Datum: 18.12.2003, Autor: J. Müller	

**Abbildung 9.10.** Vorlage Teilprojekte identifizieren

9.11 Prozessschnittstellen aufnehmen

Ziel:

Identifizieren aller Prozessschnittstellen zu einem Zulieferprozess.

Spaltenbeschreibung:

Zulieferprozess: Name des Zulieferprozesses, für den die Schnittstellen betrachtet werden. In der Spalte werden die Aktivitäten aufgeführt, welche die Schnittstellen zu den angrenzenden Prozessen beinhalten.

Zulieferprozess I bis IV: Name der Zulieferprozesse, die durch die Aktivität beeinflusst werden.

Nutzen:

Kontrolle, ob alle in der Prozesslandkarte aufgeführten Schnittstellen berücksichtigt wurden.

Prozessschnittstellen aufnehmen	Projekt: Projektname				t-wfhi-11
Zulieferprozess:	<Zulieferprozess I>	<Zulieferprozess II>	<Zulieferprozess III>	<Zulieferprozess IV>	
Aktivität x					
Aktivität y					
Aktivität z					
...					
Projektleiter: Projektleiter; Datum: 06.07.2004					
Dokumentversion: 0.1, Datum: 18.12.2003, Autor: J. Müller					

Abbildung 9.11. Vorlage Prozessschnittstellen aufnehmen

## 9.12 Identifizierte Anwendungen

**Ziel:**

Überblick über die verwendeten Anwendungen und deren Funktionen.

**Vorgehen:**

Es sollen anhand der Geschäftsprozesse die Zulieferprozesse und die zur Durchführung benötigten Anwendungen inklusive der Funktionen identifiziert werden. Für jeden Geschäftsprozess soll mindestens ein Formular ausgefüllt werden.

**Spaltenbeschreibung:**

Geschäftsprozess:	Name des Geschäftsprozesses, der mehrere Zulieferprozesse enthalten kann.
Verantwortlicher Geschäftsprozess:	Name des Prozessverantwortlichen des Geschäftsprozesses
Verantwortlicher Zulieferprozess:	Name des Prozessverantwortlichen des Zulieferprozesses
Zulieferprozess:	Name des Zulieferprozesses
Anwendung:	Name der Anwendung
Funktionalität:	Beschreibung der Fachfunktionalität, die in der Anwendung verwendet wird.
Schnittstellentyp:	Auswahl des Schnittstellentyps, falls über die Fachfunktion Daten ausgetauscht werden. Zur Auswahl stehen: Dateischnittstelle, API-Aufruf (hier kann die genaue Technologie genannt werden)

**Nutzen:**

Hilft, nur die Anwendungen zu betrachten, die für den Geschäftsprozess relevant sind.

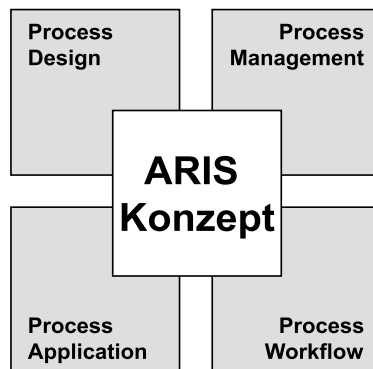


## 10 Anhang

### 10.1 ARIS-Konzept

ARIS (Architektur Integrierter Informationssysteme) ist ein Konzept zur Beschreibung von betriebswirtschaftlichen Anwendungssystemen. Das Konzept ist in das House of Business Engineering (HOBE) eingebettet, das die Strukturen des Geschäftsprozessmanagements beschreibt und analysiert. HOBE fokussiert dabei mit den vier Phasen auf einen kontinuierlichen Verbesserungsprozess.

Die Verwendung des ARIS-Konzeptes kann unterschiedliche Zielsetzungen haben, welche aber immer die Abbildung der betriebswirtschaftlichen Realität (Ist-Situation) in einem Modell (Soll-Situation) zur Folge hat. Der Detailgrad dieser Modelle hängt stark von der Zielsetzung ab.



**Abbildung 10.1.** House of Business Engineering (HOBE)

#### 10.1.1 Phasen des HOBE

##### **Process Design (Geschäftsprozessgestaltung)**

In dieser Phase werden die Geschäftsprozesse mit den in ARIS bereitgestellten Methoden und Techniken beschrieben. Mit ihnen können Analysen, Bewertungen von Prozessen sowie Verfahren zur Qualitätssicherung durchgeführt werden, so dass eine Planung neuer und geänderter Geschäftsprozesse qualitativ und quantitativ erfolgen kann. Dadurch und durch die Entwicklung von Kennzahlensystemen zur Bewertung von Geschäftsprozessen wird die Basis für ein quantitatives Geschäftsprozessmanagement geschaffen.

**Process Management (Geschäftsprozessmanagement)**

Der Fokus dieser Phase liegt auf der Zeit- und Kapazitätssteuerung und auf einer operativen Kostenanalyse, für die Verfahren und Methoden zur Verfügung gestellt werden. Es soll dem Prozessverantwortlichen die Möglichkeit gegeben werden, die Prozesse durch prozessorientierte Controllinginstrumente unter Verwendung von operativen Daten und einem Kennzahlensystem dauerhaft zu bewerten.

**Process Workflow (Vorgangssteuerung)**

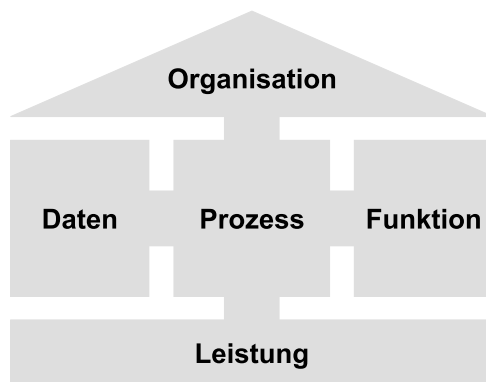
Hier wird die Bearbeitung eines konkreten Vorganges mit Hilfe einer Vorgangssteuerung durch gezielte Weitergabe von Informationen realisiert. Eine solche Steuerung kann durch die Verwendung von Workflow- oder Groupware-Systemen, aber auch durch Transportsysteme in der Industrie unterstützt werden.

**Process Application (Anwendungsausführung)**

In dieser Phase werden die einzelnen Bearbeitungen der Aktivitäten eines Geschäftsprozesses behandelt. Diese Phase kann durch bereits implementierte oder noch zu implementierende Anwendungen unterstützt werden. Bei den Anwendungen kann es sich um eine Textverarbeitung, eine Host-Lösung oder auch um ein ERP handeln.

**10.1.2 ARIS-Haus**

Aus den oben beschriebenen Sichten besteht das ARIS-Haus. Den Rahmen des Hauses bilden die vier statischen Sichten, welche durch die statisch-dynamische Prozesssicht im Inneren verbunden werden.



**Abbildung 10.2.** ARIS-Haus

Durch das ARIS-Toolset werden bereits über 100 Modelltypen angeboten, die einen Einstieg in die Modellierung vereinfachen. Es empfiehlt sich, im Vorfeld einen Modelltyp zu wählen, der für das Erreichen des Projektziels nützlich sein kann.

### 10.1.3 Objekte in ARIS

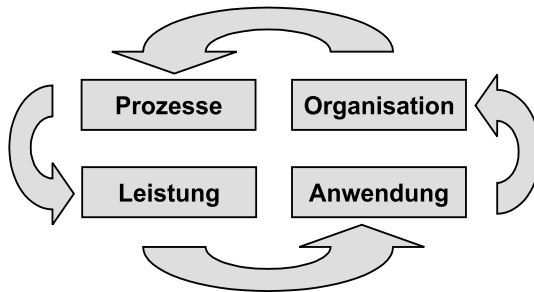
Für die Darstellung der betriebswirtschaftlichen Realität und das zu erstellende Modell sind Objekte definiert, mit denen betriebswirtschaftliche Abläufe modelliert werden können. Die folgenden Objekte stehen in einer engen Beziehung zueinander und orientieren sich in ihrer Struktur an dem ARIS-Haus.

**Tabelle 10.1.** ARIS-Objekte

Objekt	Beispiel
Funktionen	Rechnung prüfen, Ware annehmen
Daten	Kundenstammdaten, Rechnungsdaten, Stücklisten
Organisationseinheiten	Personalabteilung, Vertrieb, Produktion
Ereignisse	Ware ist eingetroffen, Rechnung ist korrekt
Ressourcen	Papier, EDV, Personal
Leistungen	Platine (produzierte Hardware), allg. Dienstleistung

Durch die enge Beziehung der Objekte untereinander entstehen folgende Abhängigkeiten:

- Ereignisse aktivieren Funktionen
- Funktionen erzeugen Ereignisse
- Organisationseinheiten sind fachlich verantwortlich für Funktionen
- Daten sind Input für Funktionen
- Leistungen sind Output von Funktionen

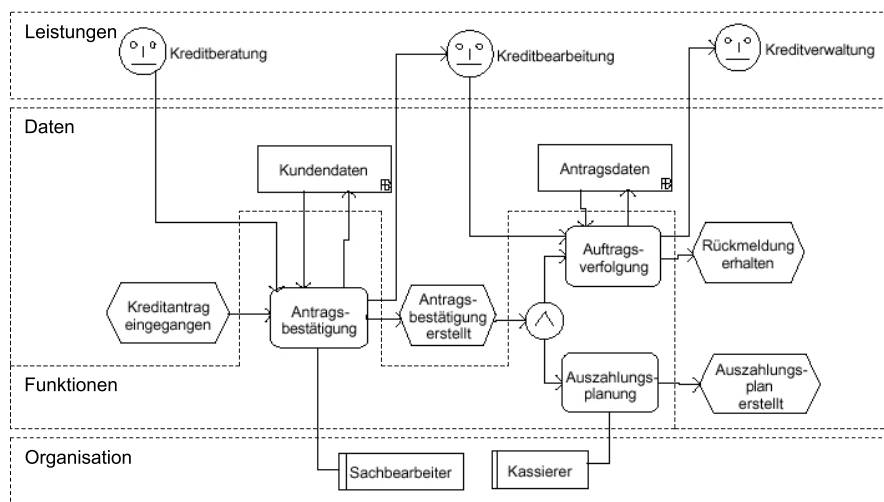


**Abbildung 10.3.** Zusammenhänge zwischen den Objekten

### 10.1.4 Sichten in ARIS

Da in einem Vorgang alle Auswirkungen auf alle Elemente des Prozesses betrachtet werden müssen, verliert das Modell sehr schnell an Übersichtlichkeit. Um dem entgegenzuwirken, wurde das Modell in vier statische Sichten (Leistungssicht, Datensicht, Funktionssicht, Organisationssicht) und eine statisch-dynamische Sicht (Prozesssicht) zerlegt. Die statisch-dynamische Sicht wird benötigt, um die

durch die Zerlegung verloren gegangenen Zusammenhänge der einzelnen Sichten wieder herzustellen. Mit der Prozesssicht werden also die Verbindungen zwischen den Sichten beschrieben. Mit der Leistungssicht werden alle materiellen und immateriellen Input- und Output-Leistungen strukturiert. In der Datensicht werden die Informationsobjekte und deren Attribute beschrieben. Da die Informationsobjekte auch Beziehungen untereinander und zu Ereignissen haben, werden diese auch in der Datensicht dargestellt. Die Ereignisse geben Auskunft über den aktuellen Status (Zustand) des Prozesses. Die Funktionssicht beschreibt die Vorgänge, die Leistungen transformieren, sowie die zwischen ihnen bestehenden statischen Beziehungen. In dieser Sicht werden auch Anwendungssysteme dargestellt, da diese die Bearbeitungsregeln der Tätigkeiten vorschreiben. In der Organisations-sicht ist die Darstellung der Elemente einer Aufbauorganisation und deren Beziehungen enthalten – zu dieser Sicht gehören auch die Betriebsmittel und die Unterstützung durch die EDV.



**Abbildung 10.4. ARIS-Sichten**

Der Vorteil dieser Zerlegung liegt in der Reduktion der Komplexität der Geschäftsprozesse. Es kann so jede Sicht einzeln bearbeitet und übersichtlich dargestellt werden. In einem weiteren Arbeitsgang werden die vier Sichten durch die Prozesssicht verbunden. Redundanzen werden vermieden, da die Elemente aus den vier Sichten in der Prozesssicht verwendet werden.

## 10.2 WfMC-Dokumenten-Index

Es soll hier ein Überblick gegeben werden, welche Themen durch welche Dokumente geführt werden. Die Aufstellung und der Status der Dokumente ist der Stand vom Juli 2004.



**Tabelle 10.2. Dokumenten-Index der WfMC**

<i>Nummer</i>	<i>Name</i>	<i>Vers</i>	<i>Status</i>	<i>Ausgabedat.</i>
WfMC-TC-1001	Technical Document Standards 2.0	2.0	abge- nommen	18. Okt. 96
WfMC-TC-1002	Document Index	2.0	abge- nommen	18. Okt. 96
WfMC-TC-1003	Workflow Reference Model	1.1	veröffentl.	19. Jan. 95
WfMC-TC-1008	Interoperability White Paper	1.0	veröffentl.	April 95
WfMC-TC-1009	Client Application API Specifications (WAPI)	2.0	veröffentl.	Juli 98
WfMC-TC-1010 <sup>17</sup>	Workflow Definition Read/Write APIs		verschoben	
WfMC-TC-1011	Terminology & Glossary	2.0 3.0	veröffentl. Entwurf	Juni 96 Feb. 99
WfMC-TC-1012	Workflow Interoperability – Abstract Specifications	1.1 2.0	veröffentl. Entwurf	29. Sep. 98
WfMC-TC-1013	WAPI Naming Conventions	1.0	veröffentl.	Nov. 95
WfMC-TC-1014 <sup>18</sup>	Applications Invocation Interface	0.9	Superceded	17. Juli 96
WfMC-TC-1015	Audit Data Specifications	1.1	veröffentl.	Sep. 98
WfMC-TC-1016	Workflow Process Definition Interchange P: Process Model X: Questions & Answers	7.05	veröffentl. Entwurf	Nov. 98 Feb. 99
WfMC-TC-1017	Conformance Testing White Paper	1.0	Beta- Entwurf	Sep. 96
WfMC-TC-1018	Interoperability – Internet E-mail MIME Binding	1.1	veröffentl.	Sep. 98
WfMC-TC-1019	Workflow Security Considerations – White Paper	1.0	veröffentl.	Feb. 98

<sup>17</sup> Dieses Dokument wurde in WfMC-TC-1009 v2.0 mit übernommen<sup>18</sup> Dieses Dokument wurde in WfMC-TC-1009 v2.0 mit übernommen

<i>Nummer</i>	<i>Name</i>	<i>Vers</i>	<i>Status</i>	<i>Ausgabedat.</i>
WfMC-TC-1020	Resource Model		nicht verfügbar	
WfMC-TC-1021	Interoperability Proving Framework document	1.02	Entwurf	Dez. 98
WfMC-TC-1022	Administrative functions	0.1	Entwurf	Okt. 98
WfMC-TC-1023	A Common Object Model – Discussion Paper	0.2	2. Entwurf	Jan. 98
WfMC-TC-1024	Process Definition Attributes List		nicht verfügbar	
WfMC-TC-2101	OMG Submission – Workflow Facility Specification RFC	1.0	veröffentl.	Sep. 97
WfMC-TC-2102	Interworkflow Application Model: The Design of Cross-Organizational Workflow Processes and Distributed Operations Management	1.0	veröffentl.	März 97

# Glossar

## **Ablauforganisation**

Beinhaltet die Durchführung von Aufgaben sowie die Koordination der zeitlichen und räumlichen Aspekte der Aufgabendurchführung.

## **Aktivitäten**

Sind die kleinste Unterteilung in einem Prozess.

## **API (Application Programming Interface)**

Eine von einem Betriebssystem oder einem Anwendungsprogramm vorgegebene Schnittstelle, über die anderen Anwendungen standardisierte Software-Werkzeuge zur Verfügung gestellt werden. Zweck eines API ist es, durch das Angebot von vordefinierten „Programm-Bausteinen“, wie etwa Routinen für die Grafik, die Ein- und Ausgabe sowie den Datenaustausch, die Programmierung neuer Anwendungen zu vereinfachen. Hierdurch wird nicht nur dem Programmierer die Arbeit erleichtert, sondern auch gewährleistet, dass eine für ein bestimmtes Betriebssystem oder Programm entwickelte Anwendung eine möglichst einheitliche Benutzeroberfläche und Funktionsweise erhält.

## **Arbeitsschritt**

Beinhaltet alle Aktivitäten, die an einem Arbeitsplatz durch eine Rolle bearbeitet werden können.

## **ASP (Application Service Provider)**

Der ASP ist ein Dienstanbieter, der in einem Kommunikationssystem (Internet) seinen Kunden anwendungsbezogene Dienste (Applikationen) bereitstellt.

## **Aufbauorganisation**

Beinhaltet die Gliederung des Unternehmens in Teilsysteme und die Zuordnung von Aufgaben zu diesen Teilsystemen.

## **Bearbeitungszeit**

Die Zeit, die für die Durchführung einer Aktivität benötigt wird.

## **BPEL (Business Process Execution Language for Web Services)**

Sie basiert auf Web Services (XLANG, WSFL, WS-Coordination, WS-Transaction). Konkurriert mit BPML.

**BPML (Business Process Modelling Language)**

BPML ist eine Metasprache, die auf XML basiert. Konkurriert mit BPEL.

**Browser**

Ist ein Programm, das von verschiedenen Herstellern bezogen werden kann, um die Masken einer webbasierenden Software anzeigen zu können. Der Browser hat die Aufgabe, die vom zentralen Server abgesendeten Seiten (Masken) auf den Rechnern des Anwenders darzustellen.

**CLR (Common Language Runtime)**

Die CLR von .NET vgl. Abschnitt 3.3.2 Verteilte Anwendungen mit Microsoft .NET.

**Compiler**

Ein Compiler übersetzt Quellcode, den Text, mit dem ein Programm geschrieben wurde, in Maschinensprache, also in eine Folge von Zahlen, die Instruktionen für die CPU liefern. Viele Computersprachen können kompiliert werden; hierzu zählen unter anderem C, C++, Pascal und Basic.

**CMS (Content Management System)**

Ein CMS ist eine Software zur Verwaltung des Inhalts einer Website. CMS wird oft mit Portal-Systemen verwechselt, aber Portale haben vor allem die Aufgabe, das Zusammenspiel zwischen den Benutzern und der Website zu steuern. CMS automatisieren den Lebenszyklus von Web-Inhalten mit dem Ziel einer effizienteren und effektiveren Herstellung, Pflege und Wartung von Web-Sites (WCMS).

**CPU (Central Processing Unit)**

Die CPU (Prozessor) ist die zentrale Rechen- und Steuereinheit eines Computers. Sie besteht aus einem oder mehreren Mikroprozessoren (Chips), die die Befehle der Programme interpretieren und ausführen. Die Leistungsfähigkeit (Performance) eines Rechners ergibt sich unter anderem aus der Geschwindigkeit, mit der seine CPU Rechenoperationen ausführen kann.

**CTS (Common Type System)**

Durch das CTS ist die CLR leistungsfähig und universell einsetzbar. Aufgrund der Objektorientierung statet das CTS jede Programmiersprache, die auf der .NET-Plattform verwendet werden kann, mit Datentypen aus. Das CTS unterstützt prozedurale und funktionale Sprachen.

**Debugger**

Der Debugger ist ein spezielles Programm, mit dem eine Ablaufverfolgung des ausgeführten Programmes in einzelnen Schritten möglich ist. Der Debugger ist in der Regel Bestandteil einer Programm-Entwicklungsumgebung (auch IDE genannt).

**EJB (Enterprise Java Beans)**

Enterprise JavaBeans sind Komponenten für die Erstellung von verteilten Business-Anwendungen. Mit EJB-Technologie erstellte Anwendungen sind multiuserfähig, skalierbar, plattformunabhängig und transaktional. Es sind Java-Programme und Bibliotheken, die Geschäftslogik enthalten und im Rahmen des J2EE (Java 2 Enterprise Edition) verwendet werden.

**EJB - Entity Bean**

Die EJB steht für den Zugriff auf permanente Daten zur Verfügung. Eine EJB-Instanz wird von mehreren Clients verwendet und entspricht in der Regel einem Datensatz in einer Datenbanktabelle.

**EJB - Message Driven Bean**

Der Aufruf der EJB erfolgt bei Eintreffen einer Nachricht über den Java Message Service. Die Daten sind wie bei der Session Bean nicht permanent und es wird kein interner Zustand verwaltet.

**EJB - Session Bean**

Die Art der EJBs stellt die Dienste zur Verfügung und kann in zwei Varianten (Stateless, Stateful) erstellt werden. Die Daten der Session-Bean sind nicht permanent, d.h. nach einem Reboot des Servers stehen die Daten nicht mehr zur Verfügung. Eine Stateful Session Bean kann, im Gegensatz zu den Stateless Session Beans, die Daten zwischen zwei Methodenaufrufen speichern.

**ERP (Enterprise Resource Planning)**

Ein Programm für die Planung und Steuerung der gesamten Wertschöpfungskette eines Unternehmens. Es besteht aus mehreren Applikationen, die dem Einkauf, der Materialwirtschaft, der Produktionsplanung und Produktionssteuerung, der Lagerverwaltung, der Personalverwaltung, der Qualitätssicherung und dem Finanzmanagement dienen.

**Framework**

Im Gegensatz zur Programmbibliothek besteht ein Framework zusätzlich aus einem Hauptprogramm, das die globale Steuerung übernimmt. Wörtlich übersetzt bedeutet Framework (Programm-) Gerüst, -Rahmen oder -Skelett. Die Grobarchitektur ist bereits vorgegeben und es kann an ganz bestimmten Stellen applikationsspezifischer Code implementiert werden. Daraus folgt, dass ein Framework eine Standard-Softwarearchitektur definiert. Die eigentliche Applikation umfasst also kein Hauptprogramm mehr und wird von Framework-Komponenten aus aufgerufen.

**Geschäftsprozess**

Ein Geschäftsprozess ist als Arbeitsablauf zu verstehen, der aus einer Folge von Aktivitäten und Ereignissen besteht. Der GP hat Ziele für den Kunden und/oder für das Unternehmen

**HTTP (Hypertext Transfer Protocol)**

Es ist ein Kommunikationsprotokoll zwischen Webserver und Webbrowser zur Übertragung von HTML-Daten und baut auf das Internet-Protokoll TCP/IP auf.

**HTTPS (Hyper Text Transfer Protocol Secure)**

Dies ist die sichere Variante von HTTP, die im WWW eine verschlüsselte Datenübertragung zwischen Browser und Server ermöglicht. HTTPS nutzt den SSL-Standard (Secure Socket Layer), einen von Netscape entwickelten, offenen Standard zur gesicherten Übertragung.

**J2EE (Java 2 Enterprise Edition)**

J2EE ist ein Standard, um mit modularen Komponenten verteilte, mehrschichtige Anwendungen zu entwickeln. J2EE setzt auf bereits etablierte Standards wie z.B. JDBC oder CORBA auf und ermöglicht dem Entwickler den Zugriff auf weitere Funktionalitäten wie z.B. Enterprise Java Beans, Java Servlets, JSP und XML.

**Java**

Java ist eine objektorientierte Programmiersprache. Aus den Quelltexten wird durch einen Compiler ein plattformunabhängiger Zwischencode übersetzt. Dieser kann von einem geeigneten Interpreter (= Übersetzer) auf beliebigen Rechnern abgearbeitet werden. Dadurch können Java-Programme auf allen Rechnerplattformen laufen, für die ein passendes Interpreterprogramm existiert.

**JDBC-Konnektor**

JDBC (Java Database Connectivity) ist eine API der Java (Programmiersprache)-Plattform, die eine einheitliche Schnittstelle zu Datenbanken verschiedener Hersteller bietet und speziell auf relationale Datenbanken ausgerichtet ist.

**JMS (Java Message Service)**

Java Message Services sind Klassen und Schnittstellenbeschreibungen für die Entwicklung nachrichtenbasierter Anwendungen.

**JSP (Java Server Pages)**

Die Java Server Pages ermöglichen die Kombination von Daten und Stylesheets zu HTML oder XML. JSP hat gegenüber von Active Server Pages (ASP) den Vorteil der besseren Trennung von Daten, Logik und Präsentation. Die Logik wird durch Servlets bereitgestellt.

**LDAP (Lightweight Directory Access Protocol)**

Das LDAP ist ein Protokoll für den Zugriff auf Verzeichnisse und definiert einen Standard für die Kommunikation mit Datenbanken im Internet. Außerdem ist es ein umfassendes Konzept zur Speicherung von und zum Zu-

gang zu Daten verschiedenster Art (z.B. Personen, Dateien, Computer, Adressen usw.). LDAP baut dabei nicht auf Tabellen auf, wie dies bei relationalen Datenbank-Management-Systemen der Fall ist, sondern bildet natürliche Strukturen ab – diese folgen häufig einem hierarchischen Ordnungsprinzip.

**Liegezeit**

Der Zeitraum zwischen der Bearbeitung von Aktivitäten ohne die Transportzeit.

**Linker**

Der Linker setzt die mit einem Compiler compilierten Objectcodes zu einem lauffähigen Programm zusammen. Damit das Programm lauffähig wird, setzt er einen vom Betriebssystem abhängigen Startercode ein. Zusätzlich bindet der Linker die benötigten Programmbibliotheken in das Programm.

**Middleware**

Software mit Schnittstellencharakter, die als Zwischenschicht zwischen Server- und Client-Komponenten eines verteilten Systems sitzt. Für den Anwender ist sie in der Regel unsichtbar, wenn verschiedene Anwendungen, Computer- oder Betriebssysteme z.B. mit Servertechniken verbunden werden.

**MS Access**

Name für ein auf Windows basierendes Datenbankprogramm der Firma Microsoft.

**Netzwerktopologie**

Die Netzwerktopologie beschreibt die Art der physikalischen Verbindung zwischen den einzelnen Computern in einem Netzwerk. Man unterscheidet zwischen:

- Bustopologie – alle Computer sind an ein zentrales Kabel angeschlossen.
- Ringtopologie – jeder Computer hat nur eine Verbindung mit seinen beiden „Nachbar“-Computern.
- Sterntopologie – alle Computer sind mit (und über) einem zentralen Knotenpunkt (HUB).
- Verbundene Baumstruktur – Struktur, in der die Computer auf verschiedenen Hierarchie-Stufen vernetzt werden.
- Vermaschte Struktur – jeder Computer hat Verbindungen zu verschiedenen anderen Computern (A ist mit B und C verbunden, C mit D und E, B mit E und D, D mit B und C usw.). Das Internet hat eine solche vermaschte Struktur.

**Open-Source-Projekte**

Open Source steht für Programme, deren Quellcode veröffentlicht worden ist. Das erste große Open-Source-Projekt war Unix. Ein weiteres Beispiel ist Linux. Nach der Definition der Opensource.org umfasst dieses Konzept allerdings noch weitere Kriterien für die Lizenzierung von Softwarenutzung. Darunter fallen zum Beispiel das Wegfallen von Beschränkungen bezüglich der Veränderung, der Anwendungsbereiche und der Weiterverbreitung.

**proprietäre Software**

Programme, deren Quellcode geschützt und unter Copyright gestellt ist. Gegenstück zu Open-Source.

**Proxies**

Ein Proxy-Server ist ein Rechner in einem lokalen Netzwerk (LAN), der bereits einmal aus dem Internet abgerufene Web-Seiten zwischenspeichert.

**Prozess**

Ein Prozess ist ein aktives Programm. Beim Start erhält er eine eindeutige Nummer. Er hat dann einen eigenen Speicherraum, einen CPU-Zustand und ist nach außen hin vom Betriebssystem abgesichert.

**Prozessinstanz**

Ist eine individuelle Kopie eines Prozesses, die einem Geschäftsvorfall entspricht.

**Prozesslandkarte**

Grafische Darstellung aller Geschäftsprozesse zur Darstellung der Zusammenhänge zwischen den Prozessen.

**IIOP (Internet Inter ORB Protocol)**

Es handelt sich dabei um ein in CORBA definiertes Protokoll auf der Basis von GIOP, mit dem ORBs über das Internet kommunizieren können, um Methodenaufrufe von Objekten auf anderen Rechnern durchzuführen.

**Runtime-Umgebung**

Besteht in der Regel aus mindestens einer ausführbaren Datei und stellt die Verbindung zwischen der erstellten Anwendung und dem Betriebssystem dar.

**Servlets**

Java-Programme, die auf einem Server ablaufen. Sie benötigen dazu eine Java-Laufzeitumgebung. Typische Umgebungen hierfür sind Apache Tomcat, Macromedia JRun oder Application-Server wie IBM Websphere oder Bean Weblogik.



**SOAP (Simple Object Access Protocol)**

Das SOAP ist ein plattformunabhängiges, XML-basiertes Protokoll, welches dazu dient, Anwendungen über das Web oder in heterogenen Computernetzen mittels des Hypertext Transfer Protocol (HTTP) miteinander kommunizieren zu lassen.

**Transaktionsverwaltung (Software)**

Eine spezielle Variante der gesicherten Datenverarbeitung. Eine Transaktion bezeichnet eine zusammenhängende Folge von Bearbeitungsschritten, die entweder komplett oder gar nicht ausgeführt werden.

**Transportzeit**

Der Zeitraum, um die Daten eines Prozesses von einer Aktivität zur nächsten zu transportieren. Bei der Verwendung von WfMS geht diese Zeit gegen null.

**WAP (Wireless Application Protocol)**

WAP ist ein Protokoll, das ähnlich wie HTTP die Kommunikation und Darstellung auf mobilen Geräten wie Handys regelt. Das WAP verbindet Mobilfunknetze mit dem Internet. Informationen werden so aufbereitet, dass sie auf dem Display WAP-fähiger mobiler Endgeräte (Mobiltelefone, PDA) dargestellt und gelesen werden können. Das Wireless Application Protocol ist die Grundlage für alle mit der Wireless Markup Language (WML) erstellten Seiten, vergleichbar mit dem HTTP für HTML-Seiten. Wegen der geringen Datenübertragungsraten und des kleinen Displays sind konventionelle HTML-Seiten für die Darstellung auf dem Mobiltelefon wenig geeignet.

**Wertschöpfungskette**

Die Kette, die den Weg eines Rohstoffs von seiner Lagerstätte bis zum Verbraucher mitsamt der in jeder Stufe erfolgten Wertsteigerung (Mehrwert) nachvollzieht, wird Wertschöpfungskette genannt (auch logistische Kette oder Supply Chain). Wenn eine Stufe mehrere Vorgänger und Nachfolger hat – was überwiegend der Fall ist –, spricht man von einem Wertschöpfungsnetz.

**Wizards**

Ist ein unterstützendes Programm, um Abläufe leichter zu erlernen oder z.B. Grafiken zu erstellen.

**Workflow-Engine**

Standardterminologie der WFMC. Software, die einen Teil oder die gesamte Runtime-Umgebung für die Prozessausführung zur Verfügung stellt. Hierzu gehören Funktionen wie das Initiieren, Starten, Beenden und Abbrechen von Prozessen.

**WPDL (Workflow Process Definition Language)**

Die WPDL soll die standardisierte Formulierung von Arbeitsabläufen erlauben.

**XML (Extensible Markup Language)**

Die XML ist eine vereinfachte Form der SGML und Quasi-Standard zur Erstellung strukturierter Dokumente im World Wide Web oder in Intranets. XML wird erweiterbar (engl. extensible) genannt, weil es seine eigenen Auszeichnungs-Tags erstellen kann. Mit Tags werden die Daten durch Bezeichner eingefasst. XML wurde von einer Arbeitsgruppe entwickelt, die unter Schirmherrschaft des World Wide Web Consortium (W3C) steht. Die Seitenauszeichnungssprache wurde vom WWW-Konsortium empfohlen, um das Web anwenderfreundlicher zu machen.

## Literaturverzeichnis

- Oestereich, B. (1997) Objektorientierte Geschäftsprozessmodellierung mit der UML, Artikel aus [www.oose.de](http://www.oose.de)
- Oestereich, B.; Weiss, C.; Weilkiens, T.; Schröder, C.; Lenhard, A. (2003) Objektorientierte Geschäftsprozessmodellierung mit der UML, dpunkt.Verlag GmbH
- Bass, L.; Clements, P.; Kazman, R. (1998) Software Architecture in Practice, Addison-Wesley
- Balzert, H. (2000) Lehrbuch der Software-Technik, Bd.1. Software-Entwicklung, 2. Auflage, Spektrum Akademischer Verlag
- Baumgarten, B. (1990) Petri-Netze: Grundlagen und Anwendungen. BI-Wissenschaftsverlag
- CARNOT AG (2003) Administrationshandbuch Version 2.7, CARNOT AG
- Chen, R; Scheer, A.-W. (1994) Modellierung von Prozessketten mittels Petri-Netz-Theorie, Veröffentlichung des Instituts für Wirtschaftsinformatik, Heft 107, Universität Saarbrücken
- Computerwoche Studie (1999) WORKFLOW-TRENDS 2000 Unterwössen: Peschanel & Partner GmbH
- Dammeyer, F. (2000) EasyFlow Administrations Toolkit Handbuch Version 2.6, TOPAS-IT GmbH
- Damschik, I.; Häntschel, I. (1994) Workflow-Management: Produktevaluierung im Labor. Institutsbericht 94.02, Institut für Wirtschaftsinformatik der Johannes Kepler Universität, Linz
- Dandl, J. (1999) Objektorientierte Prozeßmodellierung mit der UML und EPK, Arbeitspapiere WI Nr. Nr. 12/1999, Universität Mainz
- Eberhart, A.; Fischer, S. (2003) Web Services, Grundlagen und praktische Umsetzung mit J2EE und .NET, Carl Hanser Verlag

- Grässle, P.; Baumann, H.; Baumann, P. (2000) UML projektorientiert – Geschäftsprozessmodellierung, IT-System-Spezifikation und Systemintegration mit der UML, Galileo Press GmbH, 1. Auflage
- Heilmann, H. (1994) Workflow Management – Integration von Organisation und Informationsverarbeitung, HMD 176/1994
- Herrmann, T.; Scheer, A.-W.; Weber, H. (Hrsg.) (1998) Verbesserung von Geschäftsprozessen mit flexiblen Workflow-Management-Systemen – Von der Erhebung zum Sollkonzept, Physica-Verlag
- Ide, M. (2002) J2EE-Programmierhandbuch, Einführung in die Java2 Enterprise Edition, Software&Support Verlag GmbH
- IDS Scheer AG (2003), Schulungsunterlagen: ARIS Toolset, ATS1-Version 6.2
- Keller, W. (2002) Enterprise Application Integration: Erfahrungen aus der Praxis, dpunkt.Verlag
- Koch, O.W.; Zielke F. (1996) Workflow-Management – Prozeßorientiertes Arbeiten mit der Unternehmens DV; Markt&Technik Buch- und Software-Verlag GmbH
- Kueng, P. (1995) Ein Vorgehensmodell zur Einführung von Workflow-Systemen, Institutsbericht 95.02, Universität Linz, Institut für Wirtschaftsinformatik
- Kuffner, M. (2001) Enterprise Application Integration (EAI) in der Praxis, Erfahrungsbericht, Sun Microsystems GmbH – Sun Java Center
- Linthicum, D. (1999) Enterprise Application Integration 1st Edition - Paper, Addison-Wesley
- Maurer, G. (1996) Von der Prozeßorientierung zum Workflow Management, Teil 2: Prozeßmanagement, Workflow Management, Workflow-Management-Systeme, Arbeitspapiere WI Nr. 10/1996, Universität Mainz
- Maurer, G. (1997) CORBA-basierte Workflow-Architektur, Arbeitspapiere WI Nr. 12/1997, Universität Mainz
- Metastorm (2002) Installation Guide, Version 5.3, Metastorm Deutschland GmbH
- Muth, P.; Weissenfels, J.; Weikum, G. (1998) What Workflow Technology can do for Electronic Commerce; In: Proceedings of the EURO-MED NET Conference

- OMG (1995) CORBA 2.0 Specification, OMG-Document
- Picot, A.; Rohrbach, P. (1995) Organisatorische Aspekte von Workflow-Management-Systemen, Information Management 1/1995
- Pidd, P. (2003) Tools for Thinking: Modelling in Management Science, 2nd Edition, Wiley
- Renz, B. (2003) Softwarearchitektur und Anwendungsentwicklung – Entwurf Version 0.2, FH Gießen-Friedberg
- Reichert, M. (2002) Enterprise Application Integration: Von der daten- zur prozessorientierten Kopplung von Anwendungssystemen, Universität Ulm
- Rump, F. (1999) Geschäftsprozeßmanagement auf der Basis ereignisgesteuerter Prozeßketten – Formalisierung, Analyse und Ausführung von EPKS, Teubner
- Scheer, A.-W. (1995) Wirtschaftsinformatik – Referenzmodelle für industrielle Geschäftsprozesse, Springer-Verlag
- Scheer, A.-W. (2001) Die eTransformation beginnt, Physica Verlag
- Seemann, J.; Wolff von Gudenberg, J. (1999) Softwareentwurf mit UML, Springer-Verlag
- Sinur, J.; Thompson, J. (2003) Magic Quadrant for Pure-Play BPM, M-20-0930, Gartner Research
- Sinur, J.; Thompson, J. (2003) The Business Process Management Scenario, AV-20-0932, Gartner Research
- Teufel, S.; Sauter, C.; Bauknecht, B. (1995) Computerunterstützung für die Gruppenarbeit; Addison-Wesley
- Turau, V.; Saleck, K.; Schmidt, M. (2001) Java Server Pages und J2EE, Unternehmensweite Web-basierte Anwendungen, dpunkt.Verlag GmbH
- Vossen G.; Becker J. (Hrsg.) (1996) Geschäftsprozessmodellierung und Workflow-Management, International Thompson Publishing
- Watson, E. (2000) Middleware, EAI and Workflow, ISDS 7550-7553 Project Report
- Workflow Management Coalition (1995) The Workflow Reference Model, Document Number TC00-1003, Document Status - Issue 1.1

# Stichwortverzeichnis

- Aktivitätenliste 23
- ARIS-Konzept 219
  - ARIS-Haus 220
    - Objekte 221
    - Sichten 221
- Bus-oriented* 53
- CORBA
  - IDL-Compiler 73
  - IDL-Skeleton 73
  - Object Request Broker 72
- Distributed Objects 54
- Dokumenten-Management-System 15
- Dynamische Elemente 90
- EAI 35
  - A2A-Integration 36
  - Adapter 46
  - B2B-Integration 36
  - B2C-Integration 37
  - Benutzerschnittstellenintegration 41
  - Datenintegration 38
  - Debugging 50
  - Dialogintegration 41
  - Funktionsintegration 38
  - Glueware 39
  - Integrationsarten 38
  - Kommunikationsarten 45
  - Komponentenintegration 39
  - Middleware 55
  - Monitoring 50
  - Nachrichtenformate 46
  - Namensdienst 48
  - Prozessintegration 41
  - Recovery 50
  - Repository 48
  - Routing 48
  - Sicherheit 47
  - Skalierbarkeit 51
  - Tracing 50
  - Verteilbarkeit 51
- EAI-Architekturen 51
- E-Mail-System 15
- Entwicklungsumgebung 146
- EPK
  - Ereignisse 95
  - Funktionen 95
  - Informationsobjekt 96
  - Organisationseinheiten 95
  - Prozesswegweiser 96
- Ereignis 7
- Groupware-Applikationen 15
- Hub & Spoke 53
- J2EE-Architektur
  - Business Layer 66
  - Client Layer 66
  - Date Layer 66
  - EJB 66
  - EJB-Container 68
  - Enterprise-Information-System 66
  - Entity Bean 68
  - J2EE-Rollen 69
  - Message Driven Bean 69
  - Session Bean 68
  - Web Layer 66
  - Web-Container 68
- JSP-Container 41
- Load Balancing Siehe EAI
  - Hot Standby 49
  - Standby 49
- Messkriterien für Geschäftsprozesse 132
- Middleware
  - Kategorien Siehe EAI
- Modellierung 79
  - Aktivität 139
  - Datenfluss 87
  - Ereignis 139
  - Ereignisgesteuerte Prozessketten 94
  - fünf Ebenen 79
  - Geschäftsprozessebene 80
  - Kontrollfluss 87, 96
  - Materialfluss 97
  - Meta-Ebene 79
  - Modell 82
  - Modellierungsmethoden 83, 86
  - Modellierungsregeln 98
  - Nebenläufigkeit 86

- Prozessverantwortliche 85
- UML 99
- Vorgangsebene 80
- Zustand 139
- Modellierungskomponente 11
- Modellierungsmethoden
  - Petrinetze 87
- Open-Source 162
  - Lizenzmodelle 163
- Point-to-Point-Integration 52
- Produkte 167
  - Auswahlkriterien 145, 190
  - Eigenentwicklung 195
  - Entscheidungsmatrix 190
  - Kriterien 167
  - Modellierungswerkzeuge 198
- Projekte
  - Anfangsbedingungen 130
  - Freigabe 150
  - Implementierung 148
  - Laufzeitdaten 151
  - Modellierung 135
  - Optimierung 150
  - Phasen 129
  - Projektorganisation 152
  - Projektziel 133
  - Test 149
  - Vorgehensmodell 127
- Projektmanagement Applikationen 16
- Prozess
  - Geschäftsprozess 7
- prozessbasierte Integration 115
- Prozessdefinition 31
- Prozesseigenschaften 27
- Prozessinstanz 12
- Prozesskontrolldaten 23
- prozessrelevante Daten 23
- Prozess-Variablen 8
- Schnittstellenkomponente 12
- Simulationskomponente 13
- Software-Architektur 59
  - .NET 75
  - CORBA 70
  - Drei-Schichten-Architektur 64
  - Erweiterbarkeit 61
  - Funktionalität 61
  - J2EE-Architektur 66
  - Model View Controller 77
  - Modularität 61
  - Performance 62
  - Schichtenmodell 62
  - Sicherheit 62
  - Skalierbarkeit 61
  - Stabilität 62
  - Verfügbarkeit 62
  - Vier-Schichten-Architektur 65
  - Zwei-Schichten-Architektur 64
- Statische Elemente 88
- Steuerungskomponente 12
- Transaktionsbasierende Applikationen 15
- Überwachungskomponente 12
- UML
  - Aktivitätsdiagramm 102
  - Anwendungsfalldiagramm 101
  - Diagrammtypen 100
  - Implementierungsdiagramm 111
  - Interaktionsdiagramm 104
  - Klassendiagramm 108
  - Kollaborationsdiagramm 100
  - Paketdiagramm 106
  - Sequenzdiagramm 100
  - Zustandsdiagramm 109
- Web Services 157
- WfBI-Architektur 117
- WfMC 17
  - Metamodell 20
- Workflow 8
  - Workflow Management 10
  - Workflow Management System 11
  - Workflow-Engine 12
  - Workflow-Typen 8
- Workflow Management Coalition 17
- Workflow Reference Model 17
  - Interface 1 19
  - Interface 2 19
  - Interface 3 19
  - Interface 4 19
  - Interface 5 20
- Workflow-Client 23
- Workflow-Regelkreis 28

Druck:           Strauss GmbH, Mörlenbach  
Verarbeitung:   Schäffer, Grünstadt