

Prüfer : Prof. Dr. rer. nat. K. Rothermel
Betreuer : Dipl. Inform. Stefan Schreyjak
Beginn der Arbeit : 15. 08. 1996
Ende der Arbeit : 14. 02. 1997
CR-Klassifikation : C.2.4, H.2.0, H.2.4, H.4.1

Diplomarbeit Nr. 1453

**Entwurf und Realisierung einer Workflowsystem-
Komponente zur Verwaltung der Aufbau-
organisation eines Unternehmens**

Dierk Wegener

Zusammenfassung

Workflow-Management-Systeme sind Gegenstand aktueller Forschung in der Wirtschaftsinformatik. Durch eine Abkehr vom arbeitsteiligen hin zu prozeßorientiertem Denken in den Unternehmen gewinnen solche Systeme zunehmend an Bedeutung. Durch Analyse und Neudefinition der Geschäftsprozesse in einem Unternehmen wird dieses nach prozeßorientierten Gesichtspunkten neu strukturiert. Workflow-Management-Systeme sind ein Werkzeug, um die neu definierten Geschäftsprozesse so weit wie möglich zu automatisieren und damit effizienter ablaufen zu lassen.

Die auf diese Weise neu definierten Prozesse, in ihrer Gesamtheit auch als Ablauforganisation bezeichnet, laufen aber stets vor dem Hintergrund der Aufbauorganisation ab. Die Aufgabe des Workflow-Management-Systems im Unternehmen besteht in der Zuordnung von Arbeitspaketen auf Mitarbeiter. Ohne die genaue Kenntnis der Aufbauorganisation ist diese Zuordnung nicht möglich.

Die Modellierung der Aufbauorganisation aber ist ein Schwachpunkt bestehender Systeme, da sie meist in andere Komponenten integriert wird, z.B. in eine allgemeine Ressourcenverwaltung. Die Herauslösung und Realisierung der Organisationskomponente als eigenständiges Modul ist ein Ziel dieser Arbeit. Dazu soll ein flexibles Datenmodell entworfen werden, das die Abbildung von betrieblichen Organisationsstrukturen ermöglicht. Desweiteren ist als Teil der Schnittstelle zum Workflow-Management-System eine Abfragesprache zu entwerfen, die die Spezifikation von Anforderungen, die ein Bearbeiter zur Erfüllung einer Aufgabe erfüllen muß, erlaubt. Dabei soll der Benutzer Einfluß auf die Auswahl eines Bearbeiters nehmen können. Durch die Realisierung einer separaten Schnittstelle wird die Funktionalität der Organisationsdatenbank auch anderen Anwendungen zur Verfügung gestellt.

Inhaltsverzeichnis

Zusammenfassung.....	2
1. Einleitung.....	6
1.1. Die Notwendigkeit des Wandels.....	6
1.2. Betriebswirtschaftliche Lösungsansätze.....	6
1.2.1. Lean Management.....	7
1.2.2. Business Process Reengineering.....	7
1.3. Workflow Management.....	8
1.3.1. Begriffe.....	9
1.3.2. Komponenten eines Workflow-Management-Systems.....	10
1.3.3. Arbeitsweise eines WfMS.....	11
1.3.4. Beispiel: Reisekostenabrechnung.....	11
1.4. Die Aufbauorganisation eines Unternehmens.....	13
1.4.1. Aufgabenanalyse.....	13
1.4.2. Bestandteile einer Aufbauorganisation.....	14
1.5. Die Organisationskomponente eines WfMS.....	15
1.6. Motivation für die Arbeit.....	16
2. Anforderungsanalyse.....	18
2.1. Anforderungen der Anwendungsumgebung.....	18
2.1.1. Die Situation im Unternehmen.....	18
2.1.2. Folgerungen.....	18
2.1.3. Funktionale Anforderungen des WfMS.....	19
2.2. Anforderungen der Organisationsstrukturierung.....	19
2.2.1. Theorie.....	19
2.2.2. Beispiele.....	23
2.3. Weitere Anforderungen.....	26
2.4. Grobentwurf.....	27
2.4.1. Organisationsdatenbank.....	27
2.4.2. Verwaltung der Organisationsdatenbank.....	27
2.4.3. Anwendungsschnittstelle.....	28
3. Entwurf der Organisationsdatenbank.....	29
3.1. Begriffsbildung.....	29
3.2. Das Entity-Relationship-Diagramm.....	30
3.2.1. Beschreibung der Entitätsmengen.....	32
3.2.2. Beschreibung der Relationships.....	33
3.2.3. Ergänzungen und Bemerkungen zum Datenmodell.....	34
3.2.4. Varianten.....	35
3.3. Das Datenbankschema.....	35
3.4. Beispiele.....	37
3.5. Vergleich mit einem anderen Modell.....	37
4. Die Organisationsdatenbankverwaltung.....	41
4.1. Anforderungen an das Programm.....	41
4.2. Anforderungen an den Benutzer.....	41
4.3. Entwurf des Verwaltungsprogramms.....	41
4.3.1. Modellierung der Entities.....	42
4.3.2. Modellierung der Relationships.....	42
4.4. Modellierung der Datenbank.....	42
4.5. Bedienung des Verwaltungsprogramms.....	42
4.5.1. Bedienung der Maskenoberfläche.....	43

4.5.2. Bedienung der grafischen Oberfläche.....	43
4.6. Zusammenfassung.....	43
5. Die Anwendungsschnittstelle.....	45
5.1. Anforderungen an die Anwendungsschnittstelle.....	45
5.1.1. Spezifikation eines Bearbeiters.....	45
5.1.2. Bezug auf vorhergehende Anfragen.....	46
5.1.3. Übersicht über die Suchfunktionalität.....	46
5.1.4. Erfolgreiche Suche.....	47
5.2. Formale Spezifikation der Suchfunktion.....	48
5.3. Vorgehensweise der Anwendungsschnittstelle.....	49
5.3.1. Grundalgorithmus.....	49
5.3.2. Auswertung der Parameter.....	49
5.4. Kommunikation.....	50
5.4.1. CORBA.....	50
5.4.2. IDL (Interface Definition Language).....	51
5.5. IDL-Spezifikation der Anwendungsschnittstelle.....	51
5.6. Anforderungen an die Datenbankschnittstelle.....	52
6. Die Schnittstelle der Organisationsdatenbank.....	54
6.1. Anforderungen an die Datenbankschnittstelle.....	54
6.1.1. Anforderungen des Verwaltungsprogramms.....	54
6.1.2. Anforderungen der Anwendungsschnittstelle.....	54
6.2. Funktionalität der Datenbankschnittstelle.....	54
6.3. Algorithmen.....	55
6.3.1. Grundfunktionen.....	55
6.3.2. Hilfsfunktionen.....	57
6.3.3. Beurteilung der Algorithmen	59
6.4. Kommunikation mit den Benutzern.....	60
6.4.1. IDL-Spezifikation der Datenbankschnittstelle.....	60
7. Realisierung.....	61
7.1. Mittel zur Realisierung.....	61
7.1.1. Die Programmiersprache Java.....	61
7.1.2. Das Zielsystem.....	62
7.1.3. Das relationale Datenbanksystem DB2.....	63
7.1.4. Zugriff von Java-Programmen auf relationale Datenbanken.....	63
7.2. Realisierung des Entwurfs.....	63
7.2.1. Das Verwaltungsprogramm.....	64
7.2.2. Die Anwendungsschnittstelle.....	68
7.2.4. Die Schnittstelle der Organisationsdatenbank.....	68
8. Verlauf der Arbeit und Ausblick.....	70
8.1. Ergebnisse der Arbeit.....	70
8.2. Verlauf der Arbeit.....	70
8.3. Ausblick.....	71
Literatur.....	72

1. Einleitung

1.1. Die Notwendigkeit des Wandels

Das unternehmerische Umfeld befindet sich im Umbruch. Diese Feststellung trifft nicht nur auf die wirtschaftlichen Rahmenbedingungen zu, sondern ebenfalls auf die demoskopischen, gesellschaftlichen, moralischen, politischen und sozialen Bedingungen, die für ein Unternehmen von Bedeutung sind. Die Globalisierung der Märkte sowie die Überschußproduktionen in den Basisindustrien verschärfen die Konkurrenz in erheblichem Maße; Privatisierungen großer Staatsunternehmen und die verstärkte Segmentierung bedeutender Märkte tragen zusätzlich dazu bei, den Wettbewerb zwischen Unternehmen härter zu gestalten. Die Tendenz zu immer individuelleren Kundenwünschen stellt die Unternehmen vor unübersehbare Flexibilitätsprobleme. Neue und die wachsende Verschmelzung alter Technologien schaffen neue Industriezweige und zwingen die Basistechnologien zu strukturellen Anpassungen. Alle diese Entwicklungen lassen eine Grundbedingung für das Funktionieren althergebrachter Managementkonzepte und unternehmerischer Problemlösungsstrategien vermissen: Stabilität. Da sich Entwicklungstendenzen nicht mehr oder nur noch mit großen Unsicherheitsfaktoren vorhersagen lassen, sind die Unternehmen gezwungen, Entscheidungen schnell treffen und vor allem auch schnell umsetzen zu können. In der gegenwärtigen globalen Marktsituation ist als einzige Gewißheit der Wandel geblieben; diesen Wandel müssen sich die Unternehmen zunutze machen, um eine Überlebenschance zu haben.

1.2. Betriebswirtschaftliche Lösungsansätze

Der Zwang zur schnellen Reaktionsfähigkeit auf Markterfordernisse führt zur Hinterfragung der bestehenden Entscheidungsprozesse in den Unternehmen. Damit liegt es nahe, die Prozesse in einem Unternehmen in ihrer Gesamtheit zu untersuchen, um ihre Strukturen und gegenseitigen Wechselwirkungen zu erfassen. Im Produktionsbereich hat sich diese Idee bereits in den siebziger Jahren durchgesetzt; als Ergebnis sind die in dieser Zeit entstandenen *Produktionsplanungs- und Steuerungssysteme* (PPS-Systeme) anzusehen. Neuere Ansätze sind in den achtziger Jahren mit der "Fabrik 2000" und der "Fraktalen Fabrik" des Instituts für Industrielle Fertigung und Fabrikbetrieb (IFF) der Universität Stuttgart in Zusammenarbeit mit dem Institut für Produktionstechnik und Automatisierung (IPA) der Fraunhofer-Gesellschaft für Angewandte Forschung e.V. diskutiert worden. Derartige Systeme beruhen aber auf einer feststehenden Definition der anfallenden Arbeitspakete und damit auf der starren Arbeitsteilung (Taylorismus), die für den Produktionsbereich in dieser Zeit charakteristisch war. In den frühen neunziger Jahren wurde erkannt, daß das Rationalisierungspotential im produktiven Bereich nahezu ausgeschöpft war; die Folge der starken Arbeitsteilung und dem daraus resultierenden Spezialistentum waren Motivations- und Identifikationsprobleme der in der Produktion beschäftigten Mitarbeiter, die die Rationalisierungsgewinne nahezu aufwogen.

1.2.1. Lean Management

Als Folge dieser Entwicklung wurden in der japanischen Automobilindustrie neue Produktionsstrukturen eingeführt, die als *Lean Production* weltweit Nachahmung fanden. Dieses Konzept hat die folgenden Hauptmerkmale:

- Abflachung der Hierarchien und damit einhergehend größere Nähe zwischen Management und operativer Ebene
- Stärkung der Eigenverantwortung
- Stärkung der Teamarbeit
- Erhöhung der Identifikation mit dem Unternehmen
- Qualitätsmanagement

Die Übertragung der im Produktionsbereich gewonnenen Erkenntnisse auf den gesamten Verwaltungsapparat, in dem im Gegensatz zur Produktion trotz hoher Investitionen keine Produktivitätssteigerungen erzielt werden konnten, und auf das Management hat das heute aktuelle Führungskonzept des *Lean Managements* hervorgebracht. Um den erhöhten Anforderungen an Produktivität und Qualität gerecht werden zu können, sollen die sich Mitarbeiter mit ihrem Unternehmen identifizieren und an der "gemeinsamen Sache" mitarbeiten.

1.2.2. Business Process Reengineering

Die kontinuierliche Zergliederung der einzelnen Unternehmensaufgaben, der "Taylorismus", hat dazu geführt, daß der Gesamtprozeß kaum Beachtung gefunden hat. Dadurch ergeben sich u.a. folgende Probleme:

- Die Geschäftsprozesse sind kaum bekannt und zumeist nicht dokumentiert.
- Die aktive Steuerung der Prozesse ist so gut wie nicht möglich.
- Die Mitarbeiter können nur Verantwortung für ihren "kleinen" Arbeitsbereich übernehmen.
- Die Auskunftsfähigkeit ("Wo liegt der Vorgang? Welchen Status hat er?") ist sehr eingeschränkt.

Der gravierendste Nachteil der extremen Arbeitsteilung aber ist der Verlust der Kreativität der Mitarbeiter. Dieser ist nur mit der Erfüllung einer Teilaufgabe betraut und hat weder die Möglichkeit noch das Interesse, den Gesamtzusammenhang zu überschauen. Eigenverantwortlichkeit, Kreativität und Kundenorientierung sind kaum gefragt; jeder ist lediglich für sein Teilsegment und damit auch nur für die ihm zugeordnete Funktion verantwortlich.

Daher wird der Gesamtprozeß immer stärker Gegenstand der Optimierungsbemühungen. Die existierenden innerbetrieblichen Abläufe werden analysiert und gegebenenfalls neu strukturiert; dieser Vorgang wird als *Business Process Reengineering* (BPR) bezeichnet [KOCH96]. Die vorhandene Ablauforganisation wird anhand einzelner Vorgänge bzw. Geschäftsprozesse untersucht; dabei geht es nicht mehr um die effizienteste Erfüllung einer

Teilaufgabe, sondern um die effizienteste Abwicklung eines Prozesses und um die Einbindung der Mitarbeiter, d.h. das Ziel ist die Optimierung der Gesamtprozesse. Damit wird auch deutlich, daß BPR nicht nur organisatorische Auswirkungen hat, sondern auch die Arbeitsweise, das Selbstverständnis und letztlich auch die Unternehmenskultur beeinflußt.

Als Ergebnis des BPR wird das Unternehmen neu organisiert, um die in ihm ablaufenden Prozesse optimal abzubilden; es ist nicht mehr rein hierarchisch oder funktionsorientiert, sondern prozeß- bzw. ablauforientiert aufgebaut. Die Gesamtheit der identifizierten und definierten Prozesse wird als das *Prozeßmodell* der Unternehmung bezeichnet. Dieses ist keinesfalls fix; da einer der Vorteile der prozeßorientierten Organisation einer Unternehmung gerade in der Flexibilität besteht, ist es notwendig, Prozesse den Markterfordernissen anpassen zu können.

1.3. Workflow Management

Beim Ablauf der beim BPR identifizierten und spezifizierten Geschäftsprozesse treten folgende Probleme verstärkt auf:

- Durch eine räumliche Verteilung des Unternehmens wird die zur Bearbeitung eines Vorgangs erforderliche *Kommunikation* erschwert; Dokumente müssen mit der Post geschickt werden, Mitarbeiter sind nicht erreichbar usw.
- Oft müssen Filialen eines Unternehmens trotz räumlicher Trennung zusammenarbeiten, um einen Auftrag zu erledigen, d.h. die *Kooperation* in solchen Konstellationen wird beeinträchtigt.
- Ganz allgemein muß *Koordination* von Tätigkeiten dort ausgeübt werden, wo mehrere Menschen zusammenarbeiten, um z.B. Mehrfachbearbeitungen zu vermeiden.

Ein systematisches Management von Geschäftsprozessen dient der Minimierung von Reibungsverlusten bei der Bearbeitung von komplexen Vorgängen, an denen mehrere Personen aus ggf. verschiedenen Abteilungen beteiligt sind. Solche Vorgänge erfordern ein hohes Maß an Kommunikation, Koordination und Kooperation der beteiligten Mitarbeiter. Systematisches Management von Geschäftsprozessen bedingt daher die Modellierung, Analyse und Ausführung von Geschäftsprozessen. Durch den Einsatz EDV-gestützter Workflow-Management-Systeme (WfMS) können diese Probleme gemindert werden, indem die Abarbeitung (teil-)automatisiert wird und eine ständige Kontrolle des Bearbeitungsstandes ermöglicht wird. Über das WfMS sind auch jederzeit die zur Bearbeitung eines Vorgangs erforderlichen Informationen verfügbar; da es die Bearbeitung der Geschäftsprozesse nach seiner Aktivierung aktiv steuert, indem es die einzelnen Bearbeitungsschritte auslöst und damit den Zeitpunkt der Bearbeitung beeinflußt, wird auch die Ausführungsdauer eines Geschäftsprozesses verringert. Dem Bearbeiter wird eine konkrete "Arbeitsanleitung" gegeben; Unsicherheiten, die die erforderlichen Arbeitsschritte betreffen, sind damit ausgeräumt. Dies wirkt sich positiv auf die Motivation der Mitarbeiter aus.

1.3.1. Begriffe

Da sich die Diskussion um das Workflow Management noch in vollem Gang befindet, ist die Begriffsbildung zu diesem Thema noch nicht einheitlich. Dennoch müssen gewisse Schlüsselbegriffe klar definiert werden.

- **Geschäftsprozeß (business process).** Ein Geschäftsprozeß besteht aus einer Menge von Aktivitäten, die ganz oder teilweise von ggf. verschiedenen Personen in einer definierten Reihenfolge ausgeführt werden müssen. Dabei sind verschiedene Alternativen möglich. Aktivitäten können beispielsweise sequentiell, parallel oder alternativ bearbeitet werden.
- **Workflow.** Die Workflow Management Coalition definiert den Begriff [WfMC96] folgendermaßen:
The automation of a business process, in whole or part, during which documents, information or tasks are passed from one participant to another for action, according to a set of procedural rules.
Ein Workflow ist also eine technische Sicht eines Geschäftsprozesses, gewissermaßen ein "Geschäftsprozeß auf dem Rechner".
- **Workflow-Management-System (WfMS).** Ein Workflow-Management-System ist ein technisches System, mit dessen Hilfe Geschäftsprozesse durch die Ausführung von Software, deren Ereignisfolge durch eine Prozeßdefinition vorgegeben ist, definiert, verwaltet und kontrolliert werden [JOO96].
- **Aktivität.** Eine Aktivität ist die Beschreibung eines logischen Schritts innerhalb eines Geschäftsprozesses. Dabei kann es sich um eine manuelle Aktivität, die nicht (direkt) von einem WfMS unterstützt wird, oder um eine automatisierte Aktivität handeln. Diese benötigt menschliche und/oder andere Ressourcen, um die Prozeßausführung zu gewährleisten; werden menschliche Ressourcen benötigt, so wird die Aktivität Bearbeitern zugewiesen [WFMC96]. Eine Aktivität kann auch eine Liste von Qualifikationen und Fähigkeiten enthalten, die zu ihrer Abarbeitung benötigt werden.
- **Bearbeiter (Actor, Workflow Participant).** Eine (menschliche) Ressource, die die Arbeit ausführt, die von einer Workflow-Aktivitätsinstanz repräsentiert wird. [WFMC96]
- **Rolle.** Eine Rolle bezieht sich auf eine Gruppe gleichartiger Bearbeiter, die dieselben Fähigkeiten und/oder Qualifikationen haben, die eine Aktivität zu ihrer Abarbeitung verlangt. Rollen werden im Zusammenhang mit der dynamischen Aufgabenzuteilung verwendet. Diese bestimmt, ob eine Person zur Abarbeitung einer bestimmten Prozeßaktivität geeignet ist [JOO96].

1.3.2. Komponenten eines Workflow-Management-Systems

Da der Umfang eines WfMS bisher nur andeutungsweise umrissen worden ist, soll an dieser Stelle kurz auf den Aufbau eines solchen Systems eingegangen werden.

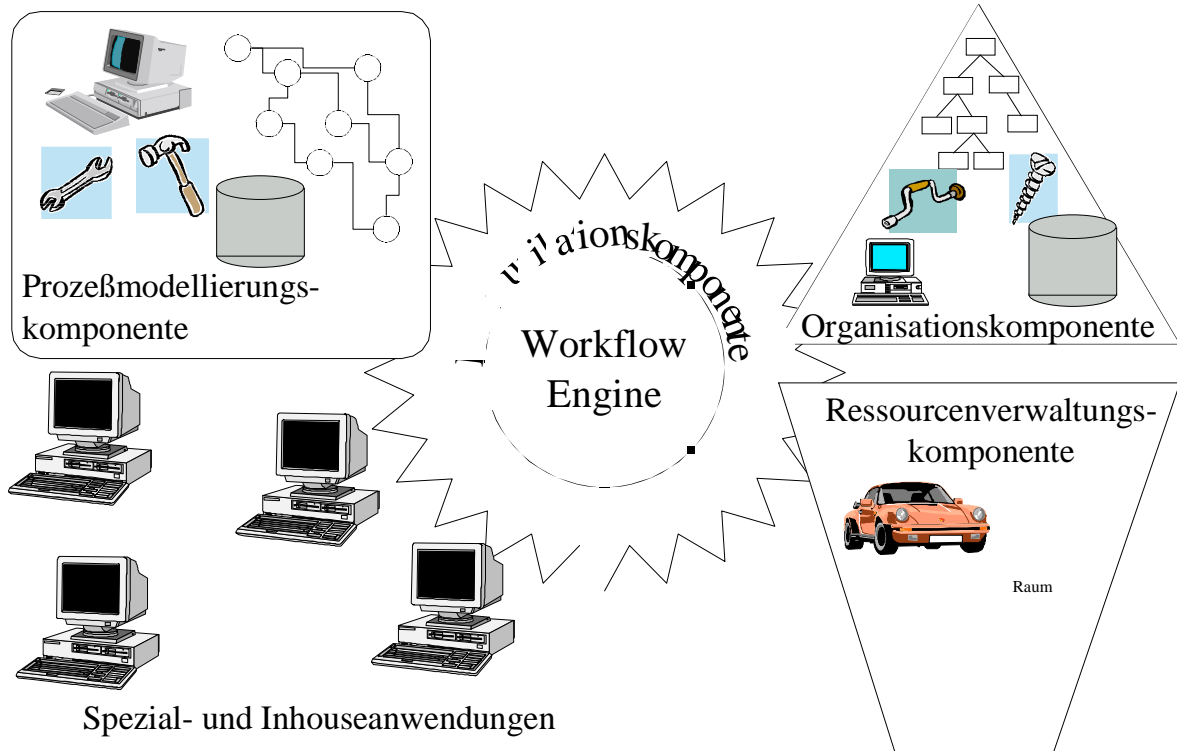


Abbildung 1.1 : Schematischer Aufbau eines WfMS

Prinzipiell umfaßt ein WfMS die folgenden Komponenten:

- **Prozeßmodellierungskomponente.** Diese Komponente ermöglicht die Eingabe und Speicherung des beim BPR erarbeiteten Prozeßmodells der Unternehmung in die EDV. Dazu stellt es Werkzeuge zur Verfügung, mit deren Hilfe Prozeßdefinitionen in das Prozeßmodell eingefügt, verändert oder gelöscht werden können.
- **Organisationskomponente.** Um eine Aktivität einem Bearbeiter dynamisch, d.h. zur Laufzeit, zuordnen zu können, muß das WfMS genaue Kenntnisse bezüglich der Aufbauorganisation des Unternehmens besitzen. Dazu gehören einerseits alle im Unternehmen vorhandenen organisatorischen Einheiten, also Stellen, Abteilungen, Bereiche usw., und ihre (festgelegten) Beziehungen zueinander, und andererseits alle konkret im Unternehmen beschäftigten Personen mit ihren Fähigkeiten und Qualifikationen. Die Organisationskomponente enthält daher ein Werkzeug zur Modellierung von Aufbauorganisationen, eine Datenbank, um das erstellte Modell zu speichern, sowie Schnittstellen, die die Verwendung der Organisationskomponente durch das System ermöglichen.
- **Ressourcenverwaltungskomponente.** Diese Komponente wird benötigt, um Unternehmensressourcen zu verwalten und im Bedarfsfall zu reservieren bzw. zu vergeben. Solche Ressourcen sind z.B. Räume oder Fahrzeuge, aber auch die Mitarbeiter eines Unternehmens.
- **Ablaufsteuerungskomponente (Workflow Engine).** Diese Komponente sorgt dafür,

daß eine Prozeßinstanz gemäß ihrer Spezifikation ausgeführt wird. Sie steuert die zeitliche Abfolge der im Geschäftsprozeß definierten Aktivitäten, so daß dem Bearbeiter einer Aktivität, falls die Aktivität einen solchen erfordert, die von ihm benötigten Informationen und andere Ressourcen jeweils zum richtigen Zeitpunkt zur Verfügung stehen.

- **Kommunikationskomponente.** Um der Ablaufsteuerungskomponente zu ermöglichen, die anderen Komponenten des WfMS anzusprechen, ist eine Kommunikationskomponente erforderlich. Sie stellt die Dienste zur Verfügung, mit deren Hilfe Mitarbeiter miteinander kommunizieren und Dokumente elektronisch verschicken können (EMail) sowie die Netzwerkschnittstellen, die für den Betrieb des WfMS in einem möglicherweise heterogenen Netzwerk benötigt werden.
- **Spezielle Verarbeitungskomponenten (Standard- oder Inhouseanwendungen).** In ein WfMS können auch externe Anwendungen wie z.B. eine Textverarbeitung, ein Projektmanagementsystem oder eine selbstgeschriebene Spezialanwendung eingebunden werden. Für deren Aktivierung ist die Ablaufsteuerungskomponente zuständig.

Aus der Architektur eines solchen WfMS geht die Möglichkeit hervor, die einzelnen Komponenten unabhängig voneinander zu realisieren.

1.3.3. Arbeitsweise eines WfMS

Die beim BPR modellierten Geschäftsprozesse mit ihren einzelnen Aktivitäten und deren Abhängigkeiten werden mit Hilfe von EDV-Werkzeugen (grafische Editoren zur Eingabe und Manipulation usw.) im System gespeichert. Wird nun ein solcher Prozeß aktiviert, so sorgt das WfMS dafür, daß jede Aktivität zum richtigen Zeitpunkt zur richtigen Bearbeitungsstelle, i.d. R. einem Bearbeiter, gelangt. Das heißt, das WfMS kontrolliert den zeitlichen Ablauf aktiver Prozesse.

1.3.4. Beispiel: Reisekostenabrechnung

Als Beispiel für einen verbreiteten, typischen (Büro-)Prozeß soll hier eine Reisekostenabrechnung, siehe z.B. [SCHR95], beschrieben werden.

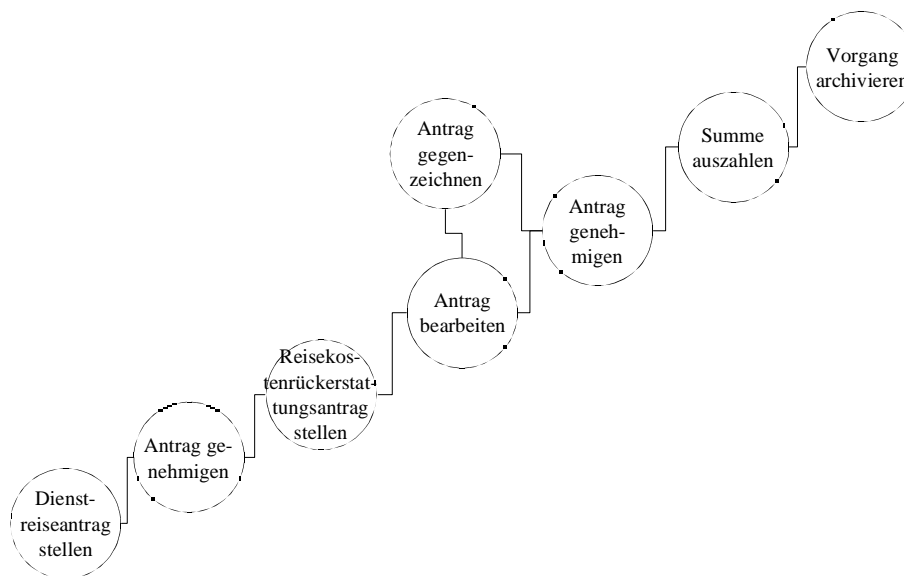


Bild 1.2: Workflow Reisekostenabrechnung

Zunächst muß sich der Mitarbeiter die Reise genehmigen lassen, d.h. er füllt einen Dienstreiseantrag aus, der seine persönlichen Daten (Name, Abteilung, Personalnummer usw.) sowie Ziel und Zweck der Reise enthalten muß. Diesen Antrag schickt er an die genehmigende Stelle, z.B. an seinen Vorgesetzten. Genehmigt dieser den Antrag, so kann der Antragsteller seine Dienstreise antreten. Während der Dienstreise sammelt er alle Belege für Spesen, die er später erstattet haben möchte. Nach seiner Rückkehr überträgt er diese Belege auf ein Formular, fügt die Originalbelege bei, trägt weitere Daten wie seine Kontonummer ein und schickt den Antrag auf Rückerstattung wieder an die zuständige Stelle. Diese prüft den Antrag, klärt Unklarheiten mit dem Antragsteller und veranlaßt im Falle der Korrektheit die Überweisung der errechneten Summe. Übersteigt diese Summe einen bestimmten Betrag, so muß zusätzlich der Vorgesetzte des Sachbearbeiters den Antrag genehmigen. Zuletzt wird der gesamte Vorgang archiviert.

Bei einer herkömmlichen Bearbeitung dieses Vorgangs können Probleme auftreten. Beispielsweise kann sich die Bearbeitung über einen längeren Zeitraum (z.B. mehrere Wochen) hinziehen, ohne daß der Antragsteller die Möglichkeit hat, den aktuellen Bearbeitungsstand oder den Verweilort seines Antrags in Erfahrung zu bringen. Die Hauspost kann lange Transportzeiten oder eine Fehlleitung verursachen. Ein Sachbearbeiter kann sich über die nächste Bearbeitungsstelle im unklaren sein und den Antrag an eine falsche Stelle weiterleiten. Ist ein Sachbearbeiter krank oder im Urlaub, so kann der Antrag über Wochen unbearbeitet liegenbleiben, wenn eine Vertretung nicht vorhanden oder nicht bekannt ist. Durch fehlende Kenntnisse über die Vorarbeit einer anderen Bearbeitungsstelle kann es zu Mehrfachbearbeitungen kommen.

Letztlich lassen sich diese Probleme auf mangelnde Kontrollmöglichkeiten während der Bearbeitung des Vorgangs zurückführen. Die Ursache dafür liegt in einer meist informellen, im Lauf der Zeit gewachsenen Spezifikation der Bearbeitungsschritte und ihrer Abfolge. In einem WfMS ist eine solche Prozeßspezifikation explizit festgehalten. Eine computergestützte Bearbeitung könnte daher folgendermaßen aussehen:

Der Antragsteller trägt reiserrelevante Daten in ein Bildschirmformular ein; dem System bekannte Daten wie Name usw. werden automatisch hinzugefügt. Das System prüft, ob das Formular vollständig ausgefüllt wurde und schickt es in diesem Fall an die für die Bearbeitung zuständige Stelle; ansonsten moniert es fehlende Daten. Wird die Reise genehmigt, so muß der Antragsteller wie bisher auf seiner Dienstreise alle Belege sammeln.

Nach der Reise füllt er den (Bildschirm-)Antrag für Reisekostenrückerstattung aus; die eingegebenen Beträge werden dabei automatisch addiert. Nach einer Plausibilitätsprüfung, und nachdem der Antragsteller seine Belege eingescannt und an das Antragsformular elektronisch geheftet hat, wird der Antrag vom System automatisch an die nächste vom Prozeßablauf spezifizierte Bearbeitungsstelle per elektronischer Post weitergeleitet. Ist der zuständige Sachbearbeiter nicht erreichbar, so wird der Antrag an einen definierten Vertreter weitergeleitet. Der Antrag wird abgezeichnet und an die Buchhaltung geschickt (wieder per elektronischer Post), wo er in eine Warteschlange gelangt, die von allen Sachbearbeitern gemeinsam abgearbeitet wird. Ein freier Sachbearbeiter entnimmt der Warteschlange den Antrag, prüft die eingescannten Belege und genehmigt die Rückerstattung. Übersteigt der Betrag eine bestimmte Summe, so wird der Antrag automatisch dem Abteilungsleiter zur Genehmigung vorgelegt. Ist der Antrag dann genehmigt, so veranlaßt das WfMS die Überweisung des Betrages auf das Konto des Antragstellers, verbucht den Betrag und benachrichtigt den Antragsteller. Die dazu notwendigen Daten sind dem System bekannt und können verwendet werden. Schließlich werden alle Formulare und Daten sowie ein Protokoll über den Vorgang archiviert. Der Bearbeitungsstand kann während des gesamten Ablaufs abgefragt werden.

Der gesamte Ablauf wird vom WfMS gesteuert, so daß eine Änderung in der Prozeßspezifikation an zentraler Stelle ohne großen Aufwand möglich ist. Will der Abteilungsleiter beispielsweise einen Reisekostenrückerstattungsantrag ab einer gewissen Summe selbst bearbeiten und nicht nur genehmigen, so wird die Prozeßspezifikation geändert; das WfMS führt den nächsten Prozeß gemäß der neuen Spezifikation aus, ohne daß Sachbearbeiter oder Antragsteller sich an neue Dienstvorschriften gewöhnen müssen.

1.4. Die Aufbauorganisation eines Unternehmens

Neben der Ablauforganisation, die durch das BPR in Form des Prozeßmodells festgelegt und mit Hilfe eines WfMS implementiert wird, spielt die Aufbauorganisation eine wichtige Rolle im Unternehmen. Mit der Aufbauorganisation eines Unternehmens ist dabei die Strukturierung der Organisationsbestandteile, der sog. organisatorischen Einheiten, gemeint. Zur Festlegung einer Aufbauorganisation wird zunächst eine Aufgabenanalyse durchgeführt; die anschließende Aufgabensynthese dient der Definition der benötigten organisatorischen Einheiten.

1.4.1. Aufgabenanalyse

Ausgangspunkt des Organisierens ist die Betriebsaufgabe, also das Betriebsziel. Diese Gesamtaufgabe wird in Teilaufgaben zergliedert, die ihrerseits weiter unterteilt werden, bis, gewissermaßen auf der untersten Ebene, eine weitere Aufspaltung nicht mehr sinnvoll erscheint. Die Tiefe der Aufgabengliederung hängt von der zu erwartenden Aufgabenverteilung ab; sie endet in der Regel bei der Berufsspezialisierung. Das Ergebnis der Aufgabenanalyse wird in Aufgabengliederungsplänen zusammengefaßt, die ihrerseits Grundlage für die Aufgabenverteilung und die Beschreibung der Aufgaben in der Stellenbeschreibung bilden.

Die oft auftretende baumartige Struktur einer Aufbauorganisation hat ihre Ursache also in der Aufteilung einer (komplexen) Unternehmensaufgabe in kleine Aufgabenpakete.

Ein Ausschnitt aus einem verfeinerten Aufgabengliederungsplan kann z.B. wie folgt aussehen:

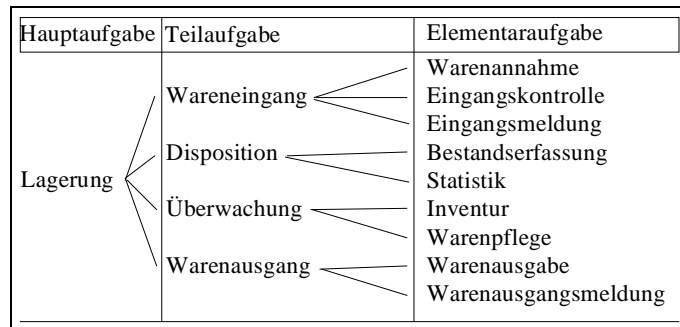


Abbildung 1.3: Aufgabengliederungsplan (Ausschnitt) [GOL83]

Abbildung 1.4 zeigt als Ergebnis einer Aufgabenanalyse die Aufgabengliederung in einem Betrieb, der elektrische Meßgeräte herstellt und vertreibt.

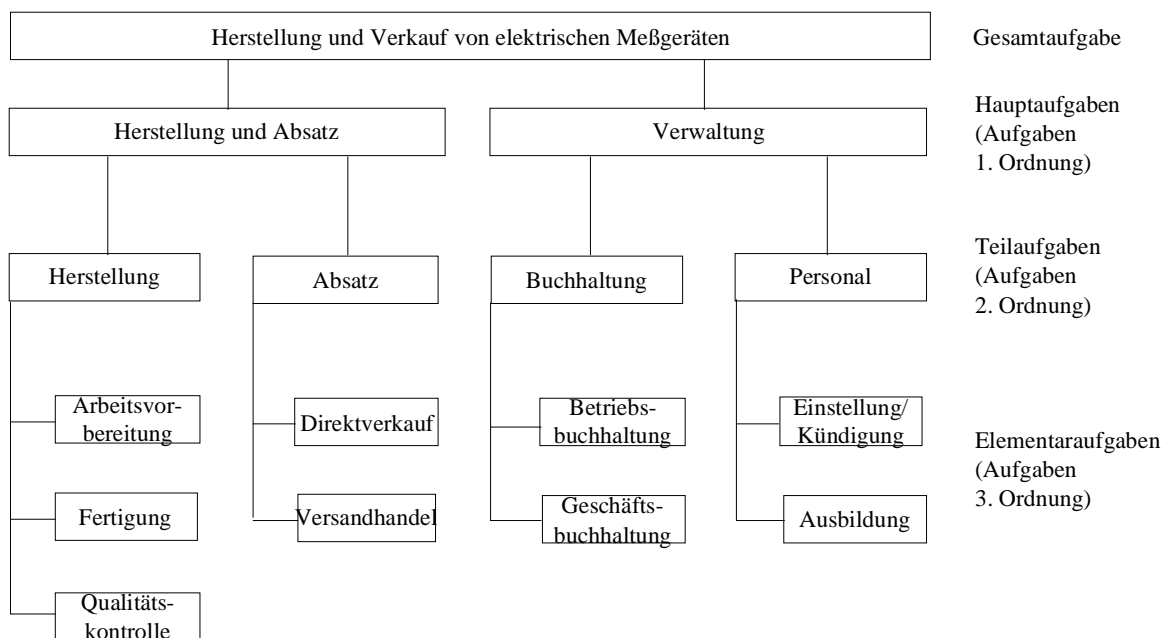


Abbildung 1.4: Beispiel für Aufgabengliederung [GOL83]

1.4.2. Bestandteile einer Aufbauorganisation

Ist nach der Aufgabenanalyse ein vollständiger Überblick über die insgesamt zu erledigenden Teilaufgaben vorhanden, so kann mit der eigentlichen Organisationsarbeit, der Zuordnung von Teilaufgaben und organisatorischen Einheiten begonnen werden. Dazu werden zunächst Aufgaben bzw. Teilaufgaben, die hinsichtlich eines Merkmals gleichartig sind (z.B. gleiches Bearbeitungsobjekt), zusammengefaßt; damit wird eine Gleichordnung geschaffen. Dieser Vorgang wird als *Aufgabensynthese* bezeichnet; dabei kommt es nicht nur zu Gleich-, sondern auch zu Über- und Unterordnungen von Aufgaben (bzw. Teilaufgaben). Damit ergeben sich vielfältige Beziehungen; es entsteht eine Struktur.

Die zusammengefaßten Aufgaben werden Menschen oder auch Maschinen zur Erledigung übertragen; da eine Unterscheidung in dieser Phase nicht sinnvoll ist, wird be-grifflich

abstrahiert von einem Aufgabenträger gesprochen. Die Zusammenfassung von Aufgaben und die Zuteilung auf einen Aufgabenträger wird als *Stellenbildung* bezeichnet; eine Stelle ist zu diesem Zeitpunkt lediglich ein Bündel von Aufgaben und Befugnissen. Die *Stelle* ist die kleinste *organisatorische Einheit*. In ihr werden die Aufgaben zusammengefaßt, die von einem (einzigem) Aufgabenträger zu erledigen sind.

Werden gleichartige Aufgabenpakete, also Stellen, zusammengefaßt, so entstehen die zusammengesetzten organisatorischen Einheiten: Gruppen, Abteilungen und Projekte. Diesen werden die entsprechenden komplexeren Teilaufgaben aus der Aufgabenanalyse zugeordnet. Damit ergeben sich bereits Beziehungen zwischen organisatorischen Einheiten; meist sind diese zunächst hierarchischer Natur. Mit der Einführung von Teams oder Projekten als Folge des Lean Managements können jedoch auch andere Beziehungen eine Rolle spielen.

Zusammenfassend läßt sich sagen: Eine Aufbauorganisation besteht aus (atomaren oder zusammengesetzten) organisatorischen Einheiten, die mit Hilfe einer Aufgabenanalyse festgelegt worden sind, und ihren Beziehungen zueinander. Sie wird meist grafisch als Organigramm dargestellt.

1.5. Die Organisationskomponente eines WfMS

Im Gegensatz zu herkömmlichen DV-Anwendungen, bei denen Strukturen, Objekte und Abläufe des Unternehmens noch implizit in den jeweiligen Anwendungen "fest verdrahtet" sind, basieren WfMS auf einer flexiblen Organisationskomponente und einem flexiblen Prozeßmodell, die die Aufbau- bzw. die Ablauforganisation des Unternehmens abbilden.

Die Organisationskomponente beschreibt die Aufbaustruktur des Unternehmens. Dazu müssen die unterschiedlichen Organisationsobjekte erhoben, analysiert, beschrieben und letztlich elektronisch abgespeichert werden. Zur Ablage dieser Informationen dient eine Datenbank, welche durch ein entsprechendes Werkzeug verwaltet bzw. manipuliert wird. Da die Aufgabe der Organisationskomponente bezüglich eines WfMS darin besteht, einer Aktivität einen "passenden" Bearbeiter zuzuordnen, wird eine Schnittstelle benötigt, über die die dazu notwendige Kommunikation abläuft. Diese Schnittstelle sollte netzwerkfähig sein, da die Komponenten des WfMS möglicherweise verteilt sind. Weiter ist zu beachten, daß das WfMS in der Lage sein muß, die Auswahl eines Bearbeiters in einem beschränkten Rahmen zu beeinflussen. Dazu kann es erforderlich sein, daß die Organisationskomponente über zugewiesene Bearbeiter Buch zu führen hat, d.h. die Anfragen und Ergebnisse derselben müssen geloggt werden.

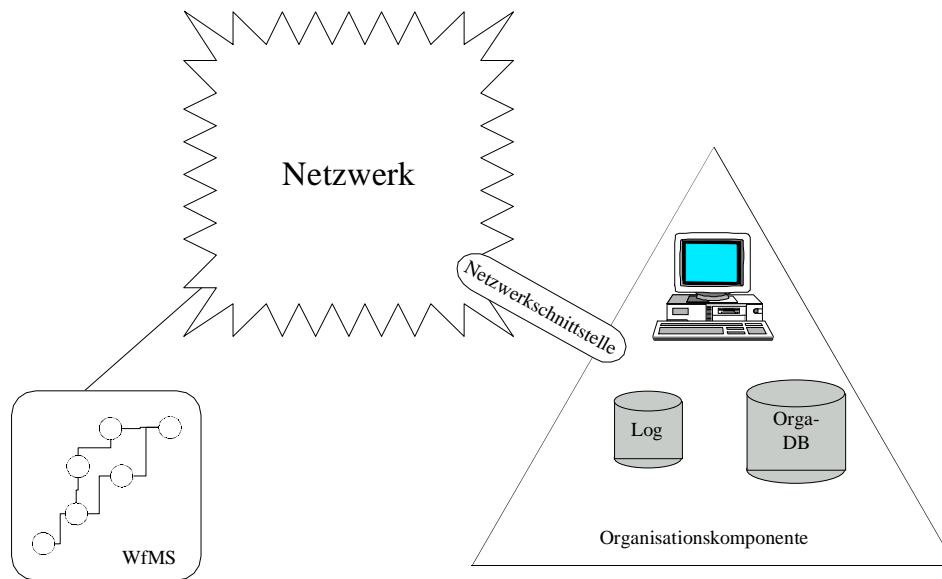


Abbildung 1.5: Grobes Schema der Organisationskomponente

Die Einführung einer flexiblen Organisationskomponente setzt eine genaue Kenntnis der vorhandenen Organisationsstrukturen voraus; ohne eine solche kann die Aufbauorganisation nicht hinreichend exakt modelliert werden. Der Vorteil einer solchen Komponente bezüglich Workflow Management besteht darin, daß die im WfMS gespeicherten Aktivitäten ohne konkreten Bearbeiterbezug verwendet werden können, da der Bearbeiter zur Laufzeit vom System ermittelt wird. Zu dieser Auswahl kann das WfMS verschiedene Suchkriterien angeben, die sich direkt auf Organisationsstrukturen beziehen können, z.B. kann der Vorgesetzte eines Bearbeiters ausgewählt werden. Ein weiterer Vorteil einer solchen Komponente besteht darin, daß die Organisationsstruktur gewissermaßen im laufenden Betrieb verändert werden kann. So können neue Mitarbeiter oder neue Gruppierungen eingetragen werden, ohne daß eine Zeile Code geändert werden müßte.

1.6. Motivation für die Arbeit

Workflow-Management-Systeme sind derzeit Gegenstand der Forschung. Existierende Systeme legen großen Wert auf die Prozeßmodellierung und -ablaufsteuerung, also auf die Ablauforganisation; die Aufbauorganisation des Unternehmens wird oft nur unzureichend modelliert. Eine Untersuchung zeigt, daß sich die Produkte in der Behandlung der Aufbauorganisation am deutlichsten unterscheiden [PIT96b].

Häufig wird die Aufbauorganisation im Rahmen der allgemeinen Ressourcenverwaltung behandelt. Da Mitarbeiter im Unternehmen tatsächlich eine (wichtige) Ressource darstellen, ist diese Vorgehensweise zwar in der Sache gerechtfertigt; sie verhindert aber oft die Verwendung der Aufbauorganisations-Wissensbasis für andere Zwecke, da in diesen Fällen nur unzureichende, meist sehr produktspezifische Schnittstellen zur Verfügung gestellt werden.

Durch die Abtrennung der Organisationskomponente von der Ressourcenverwaltung und die damit erforderliche separate Schnittstelle kann dem Benutzer eine rein aufbauorganisatorisch relevante Funktionalität angeboten werden. Als Benutzer kommt zunächst das WfMS in Betracht; es sind aber auch andere Anwendungen im Unternehmen denkbar, die als Benutzer der Organisationskomponente in Frage kommen, beispielsweise ein Teleconferencing-System.

Die Separierung der Organisationskomponente ermöglicht weiterhin deren Realisierung in einer verteilten Umgebung, beispielsweise als CORBA-Objekt.

In dieser Arbeit soll zunächst ein Überblick über die Umgebung einer Organisationskomponente gewonnen werden. Nach der Untersuchung der in Theorie und Praxis vorhandenen Organisationsmodelle und einer Anforderungsanalyse soll ein allgemeines Datenmodell entwickelt werden, mit dessen Hilfe sich die verschiedenen Organisationsobjekte und -strukturen abbilden lassen. Dieses Modell soll mit Hilfe eines Datenbanksystems implementiert werden. Desweiteren soll das Verwaltungs- und Manipulationswerkzeug dieser Organisationsdatenbank sowie die Anwendungsschnittstelle zum WfMS implementiert und getestet werden.

2. Anforderungsanalyse

Die Anforderungen, die an die Organisationskomponente eines WfMS gestellt werden, kommen aus verschiedenen Bereichen. Zunächst ist das Umfeld, in dem ein WfMS eingesetzt werden soll, beim Design von Bedeutung. Ein WfMS erwartet auch eine bestimmte Funktionalität der Komponente. Da mit dieser Komponente Aufbauorganisationen von Unternehmen modelliert werden sollen, muß zunächst untersucht werden, welche Modelle in der Organisationslehre bekannt und von Bedeutung sind. Schließlich können Anforderungen allgemeiner Art an die Komponente gestellt werden.

Als Ergebnis dieser Anforderungsanalyse soll ein erster Grobentwurf der Organisationskomponente vorgestellt werden.

2.1. Anforderungen der Anwendungsumgebung

2.1.1. Die Situation im Unternehmen

Die heterogene Infrastruktur der Informationsverarbeitung, die noch heute in Unternehmen anzutreffen ist, hat ihre Ursache in der tayloristischen Sicht des Arbeitsprozesses: Die starke Arbeitsteilung setzt sich in der DV-Unterstützung der Teilaufgaben fort, so daß in den einzelnen Bereichen Individuallösungen geschaffen wurden, die nicht miteinander kommunizieren mußten und daher dazu auch nicht in der Lage sind. Diese "Insellösungen" sind sowohl hinsichtlich der verwendeten Software als auch der Hardware meist inkompatibel zueinander.

Diese Installationen sind in den Unternehmen häufig noch in Betrieb, da ein Ersatz durch neue, modernere Systeme extrem aufwendig und damit kostenintensiv sein können. Um diese funktionierenden Systeme weiterhin nutzen zu können, zumindest übergangsweise, wurden Standards geschaffen, die die Einbindung dieser vorhandenen Hard- und Software ermöglichen, z.B. Netzwerkprotokolle wie TCP/IP oder die Netzwerkarchitektur OSI, Standard-Datenbanktreiber wie ODBC usw. Mit diesen Hilfsmitteln ist eine Koexistenz alter und neuer Systeme in der Unternehmung möglich.

2.1.2. Folgerungen

Workflow-Management-Systeme als Instrument des Business Process Reengineering sollten in der Lage sein, in einer solch heterogenen Infrastruktur zu agieren; dazu werden die angesprochenen Standards verwendet. Dies gilt ebenso für die Organisationskomponente als Bestandteil eines WfMS, d.h. die Organisationskomponente muß über Mechanismen verfügen, die einen standardisierten Zugriff auf ihre Daten auch über ein Netzwerk ermöglichen.

2.1.3. Funktionale Anforderungen des WfMS

Die Grundfunktionalität der Organisationskomponente bezüglich des WfMS besteht in der Auflösung von Anforderungen (Eigenschaften), die zur Bearbeitung einer Aktivität benötigt werden, auf einen oder mehrere konkrete Bearbeiter. Damit ist bereits das erste Problem genannt: Die Organisationskomponente muß in der Lage sein, auch mehrere Bearbeiter, falls die Aktivität dies erfordert, zu bestimmen. Möglicherweise müssen bei der Auswahl eines Bearbeiters Bedingungen gestellt oder Präferenzen gesetzt werden; auch dies muß die Organisationskomponente unterstützen. Ganz allgemein muß das WfMS die Auswahl eines Bearbeiters in gewissen Grenzen beeinflussen können.

Konkret bedeutet diese Anforderung, daß eine flexible Schnittstelle nötig ist, die sowohl die zur Auswahl eines Bearbeiters nötigen Auswahlkriterien als auch Parameter zur Beeinflussung der Auswahl zur Verfügung stellt.

2.2. Anforderungen der Organisationsstrukturierung

Keine Unternehmung gleicht im Aufbau der anderen, jede hat eine eigene, meist im Lauf der Zeit gewachsene Struktur. Die Organisationskomponente muß jedoch in der Lage sein, jede Form einer Aufbauorganisation zu speichern. Daher soll in diesem Kapitel untersucht werden, welche Arten von Aufbauorganisationen es in Theorie und Praxis gibt; da die Ablauforganisation einer Unternehmung durch die Geschäftsprozeßmodellierung im WfMS definiert wird, genügt es, sich an dieser Stelle mit der Aufbauorganisation zu befassen. Eine Aufbauorganisation besteht aus den atomaren Organisationseinheiten, den Stellen, den daraus zweckmäßig zu bildenden Gruppierungen und den Beziehungen zwischen diesen atomaren oder zusammengesetzten Organisationseinheiten.

2.2.1. Theorie

Bei der Festlegung einer Aufbauorganisation eines Unternehmens besteht die Aufgabe des Organisators darin, die gegebene Gesamtaufgabe des Betriebs (z.B. Erbringen einer Marktleistung) durch eine Aufspaltung in so viele Teilaufgaben (oder Einzelaufgaben) zu zerlegen, daß durch die anschließende Kombination dieser Teilaufgaben "eine sinnvolle Gliederung und Ordnung der betrieblichen Handlungsprozesse" entsteht [WÖ90]. Die so erhaltenen Teilaufgaben werden der elementaren Organisationseinheit in der Unternehmung, der Stelle, zugeordnet. Aus diesen Stellenaufgaben und -zielen leiten sich die mit der Stelle verbundenen Kompetenzen und Verantwortlichkeiten ab.

Stellen lassen sich nun nach unterschiedlichen Gesichtspunkten gruppieren (z.B. [SCH82]):

- **Die funktionsorientierte Organisation.** Bei einer an funktionalen Gesichtspunkten orientierten Gliederung wird auf die Kerntätigkeitsfelder von Unternehmen abgestellt. Betrachtet man z.B. Industrieunternehmen, so sind dies typischerweise die Bereiche Beschaffung, Produktion und Absatz.
- **Die divisionale oder Spartenorganisation.** Eine divisionale Gliederung setzt bei den marktorientierten Unternehmenszielen an, bei Produkten oder Dienstleistungen, Kunden oder Klienten oder gewissen räumlichen Gesichtspunkten. Wird den einzelnen Divisionen

oder Sparten die Gewinnverantwortung übertragen, so entstehen quasi-autonome Einheiten, sogenannte Profit-Center.

- **Die Matrixorganisation.** Bei der Matrixorganisation kommen mehrere Gliederungsprinzipien gleichzeitig zur Anwendung. Diese Gliederungsprinzipien müssen nicht unterschiedlich sein; sowohl eine gleichzeitige Anwendung von funktions- und markt-orientierte Gliederung als auch die gleichzeitige Anwendung zweier marktorientierter Gliederungen (z.B. Strukturierung einerseits nach Produkten, andererseits nach Regionen) ist möglich.
- **Die Projektorganisation.** Bei Projektorganisationen sind zwei Aspekte von Bedeutung. Zum einen ist die interne Struktur eines Projekts interessant, zum anderen bereitet die Eingliederung eines Projekts in eine vorhandene Aufbauorganisation Schwierigkeiten. Für die interne Strukturierung von Projekten lassen sich dieselben Unterscheidungsmerkmale nachweisen, die es für ganze Unternehmungen gibt, nämlich die Unterscheidung nach funktionaler und divisionaler Strukturierung. Auch Matrixstrukturen lassen sich finden. Bei der Einordnung eines Projektbereichs in die Gesamtorganisation geht es um die Frage, welcher Organisationseinheit die oberste Stelle des Projektbereichs unterstellt wird. Maßgeblich ist hierbei die Identifizierung der für das betrachtete Projekt zuständigen Projektinstanz. Als Projektinstanz soll die Stelle bezeichnet werden, die aufgrund ihrer Entscheidungskompetenz die Ressourcenausstattung und den Ablauf der Projektaktivitäten in dem betrachteten Projektbereich bestimmen kann.

Frese [FRE93] unterscheidet Organisationsstrukturen zunächst nach der "Dimension der Entscheidungskriterien", d.h. nach der Anzahl der Kriterien, nach denen ein Unternehmen (oder ein Teilbereich) zerlegt wird. Die Zerlegungskriterien ihrerseits lassen sich aus den Komponenten einer Entscheidungsaufgabe ableiten:

Kriterium		Beispiel	
Feld	Ressource	Personal, Material, Anlagen, Information, Kapital	
	Umwelt	Region	Region Nord, Region Süd
		Kunde	Kunde A, Kunde B
Handlung	Handlung	inhaltlicher Aspekt	Beschaffung, Produktion, Absatz
		formaler Aspekt	Planung, Kontrolle
Ziel	Produkt	Produkt(gruppe) A Produkt(gruppe) B	
	Dienstleistung	Dienstleistung A	

Abbildung 2.1: Komponenten und Dimensionen einer Entscheidungsaufgabe [FRE93]

- **Eindimensionale Organisationsstrukturen.** Eindimensionale Strukturen entstehen, wenn ein Unternehmen oder Teile davon (eine "komplexe Entscheidungsaufgabe") nach einem einzigen Kriterium in Teilaufgaben zerlegt wird, die jeweils bestimmten organisatorischen Teileinheiten einer Hierarchieebene zugewiesen werden.

- **Feldorientierte Gliederung.** Bei der Orientierung am Entscheidungsfeld der betrachteten Unternehmung werden einem Bereich alle Entscheidungen zugeordnet, die sich auf bestimmte Ressourcen (Personal, Material, Anlagen, usw) oder auf Umweltbereiche (Regionen, Kunden) beziehen. Die Verantwortungsbereiche werden in Regionen oder Märkte unterteilt.
- **Handlungsorientierte Strukturen.** Die Entscheidungen eines Bereichs sind bei der Orientierung nach diesem Kriterium auf die Festlegung gleichartiger Handlungen (z.B. Beschaffung, Produktion) ausgerichtet. Ein handlungsorientiertes Gliederungskriterium liegt der funktionsorientierten Organisationsstruktur (s.o.) zugrunde.
- **Zielorientierte Strukturen.** Alle Entscheidungen, die an der Realisierung bestimmter Sachziele, insbesondere verschiedener Komponenten des jeweiligen Leistungsprogramms, orientiert sind, werden bei Anwendung dieses Prinzips in einem Bereich zusammengefaßt. So werden in einem Produktionsbetrieb bei Realisierung des zielorientierten Prinzips die Verantwortungsbereiche nach Produkten (dem "Ziel" der Unternehmung) unterteilt.

Der Einführung mehrdimensionaler Strukturen liegt das Ziel zugrunde, die Qualität der Entscheidungen zu verbessern. Man will möglichen negativen Folgen einer gewissen Einseitigkeit eindimensionaler Strukturen bei der Problemsicht vorbeugen. Die Bildung von organisatorischen Einheiten, die an verschiedenen Dimensionen bzw. Kriterien orientiert sind, soll die Einbeziehung mehrerer Perspektiven bei der Lösung von Entscheidungsproblemen organisatorisch verankern.

- **Mehrdimensionale Organisationsstrukturen.** Mehrdimensionale Strukturen entstehen, wenn bei der Zerlegung einer komplexen Entscheidungsaufgabe auf mehr als ein Kriterium gleichzeitig zurückgegriffen wird und eine entsprechende Zuordnung auf organisatorische Einheiten erfolgt. Beispielsweise kann ein Unternehmen nach Zielen (Produkten oder Dienstleistungen) und gleichzeitig nach Funktionen (Beschaffung, Produktion und Absatz) unterteilt sein. Das Ergebnis einer solchen Gliederung sind mehrere Hierarchien im Unternehmen; u.U. hat ein Mitarbeiter mehrere Vorgesetzte, die Entscheidungen nach unterschiedlichen Gesichtspunkten fällen.
- **Stabsprinzip.** Im Rahmen des Stabsprinzips besitzen organisatorische Einheiten, die eine bestimmte Dimension, also ein Zerlegungskriterium wie Funktion, repräsentieren, keine Kompetenz über die nach mehrdimensionalen Perspektiven einzusetzenden Ressourcen. Ist z.B. eine Unternehmung nach dem Funktions- und Zielprinzip unterteilt, und ist die Dimension "Ziel" nach dem Stabsprinzip im Unternehmen angesiedelt, so hat diese Dimension nur beratenden Einfluß auf Entscheidungen, ist aber nicht weisungsbefugt. In der Praxis wird oft die EDV-Abteilung als Stabsabteilung im Unternehmen realisiert, da sie zwar von vielen Bereichen gebraucht wird, aber fachlich keine Entscheidungskompetenz haben soll.
- **Matrixprinzip.** Matrixstrukturen lassen sich auf die parallele Segmentierung eines Entscheidungskomplexes nach unabhängigen Segmentierungskriterien zurückführen. Im Unterschied zum Stabsprinzip werden alle so abgeleiteten Aktivitäten Entscheidungseinheiten zugewiesen. Die Gesamtunternehmensaufgabe oder ein spezifisches Aufgabensegment (z.B. die Absatzaufgabe) wird jeweils von mehr als einer organisatorischen Einheit aus unterschiedlichen Perspektiven (Dimensionen) bearbeitet. Die Aufgabe "Entscheidung" wird auf verschiedene Einheiten aufgeteilt; Ent-

scheidungen können so nur von den betreffenden entscheidungsbefugten Einheiten gemeinsam getroffen werden.

Zusammenfassend läßt sich sagen, daß im wesentlichen drei Hauptformen der Unternehmensstrukturierung von Bedeutung sind:

- **Die linienartige Struktur.** Alle eindimensionalen Strukturen [FRE93] wie funktionale, ziel- und feldorientierte Strukturen fallen in dieses Kriterium.
- **Die matrixartige Struktur.** Unter diesen Begriff fallen die mehrdimensionalen Strukturen nach [FRE93] wie Matrix- und Stabsorganisation. Kennzeichnend für diese Art der Aufbauorganisation ist der Umstand, daß jede Gliederungsdimension (= Gliederungskriterium) an jeder Entscheidung zumindest mitwirkt. Im Fall der Matrixorganisation ist ein explizites Miteinander der Dimensionen bei der Entscheidungsfindung erforderlich.
- **Die Projektorganisation.** Im Fall der reinen Projektorganisation kommen die herkömmlichen Arten der Unternehmensgliederung nicht zur Anwendung. Jeder Bereich im Unternehmen bildet ein Projekt. Theoretisch müßten selbst zentrale Unternehmensfunktionen wie Buchhaltung usw. als Projekt durchgeführt werden; dies ist aber wenig sinnvoll, da Projekte per definitionem Tätigkeitsbereiche temporärer Natur sind, und wird daher auch in der Realität nur sehr selten praktiziert (z.B. Ingenieurbüro). Meist ist eine Mischform aus Projektorganisation und anderen Strukturen vorzufinden.

In der Praxis sind meist Mischformen aus allen drei Bereichen vorzufinden. Die Anzahl der rein linienartig organisierten größeren Unternehmen nimmt ab; derartige Strukturen sind meist gewachsen und eher unflexibel und fallen der allgemein praktizierten Verschlan-
kung der Unternehmen zum Opfer. Große Konzerne organisieren sich oft matrixartig, da sie einerseits weltweit operieren und daher regionale Probleme zu berücksichtigen haben, andererseits aber oft große Produktpaletten haben und auch diese unterschiedlich behandeln müssen. Projekte findet man aufgrund ihrer temporären Natur hauptsächlich im Forschungs- und Entwicklungsbereich sowohl innerhalb großer Kon-
zerne als auch im universitären Umfeld oder aber in speziellen Projekt- oder Ingenieur-
büros.

Von besonderem Interesse sind die Beziehungen zwischen den Organisationseinheiten einer Unternehmung. Nach [WÖ90] stellt jedes Leitungssystem ein hierarchisches Gefüge dar, "in dem die einzelnen Stellen unter dem Gesichtspunkt der Weisungsbefugnis miteinander verbunden sind." Aber schon Stabsstellen, d.h. Stellen, die bestimmte Aufgaben übernehmen, aber keine Weisungsbefugnis, sondern nur Beratungscharakter haben, passen nicht mehr in dieses Schema. Bereits für diesen recht verbreiteten Stellentyp muß ein anderer Beziehungstyp herangezogen werden. Desweiteren werden in den Unternehmen, vorwiegend im Produktionsbereich, zunehmend "Arbeitsgruppen" gebildet. Dies hat eine Hierarchieverflachung zur Folge: Es gibt weniger Vorgesetzte und mehr "Kollegen", so daß es sinnvoll erscheint, auch dafür einen neuen Beziehungstyp vorzu-
sehen. Ein weiterer wichtiger Beziehungstyp wird durch Stellvertreterregelungen benötigt.

Eine Aufbauorganisation entsteht durch die Zergliederung der Unternehmensaufgabe bis auf "Stellenniveau", d.h. bis die entstehenden Teilaufgaben von Stellen bewältigt werden können. Daher ist eine Aufbauorganisation durch die Angabe aller definierten organisatorischen Einheiten und ihrer organisationsrelevanten Beziehungen zueinander allein noch nicht vollständig beschrieben; eine Stelle beinhaltet nicht nur eine Aufgabe, sondern auch die zu ihrer Bewältigung notwendigen Kompetenzen und Verantwortlichkeiten.

2.2.2. Beispiele

In diesem Abschnitt sollen beispielhaft Schaubilder von Organisationsstrukturen, sog. Organigramme, einiger fiktiver Unternehmungen dargestellt werden. Um das Spektrum der möglichen Organisationsformen möglichst abzudecken, soll eine linienartige, eine matrixartige, eine projektartige und eine gemischte Organisationsstruktur verwendet werden. Der Einfachheit halber sollen immer dieselben Personen verwendet werden. Diese Beispiele sollen zu einem späteren Zeitpunkt dazu dienen, die Praxistauglichkeit des gewählten Datenmodells zu überprüfen.

Folgende Namen sollen in den einzelnen Organigrammen verwendet werden:

Name	Kürzel	Persönliche Eigenschaften
Hans Scheff	HS	Dipl-Ing., Englisch, Französisch
Klaus Schwarz	KS	Dipl-Kfm., Englisch, Spanisch,
Hilde Weiß	HS	Dipl-Kfm. ¹ , Englisch, PC-Kenntnisse
Sebastian Blau	SB	DV-Kaufmann, Programmierkenntnisse
Jörg Schwarz	JS	Dipl.-Ing., Netzwerkkenntnisse
Holger Miesmann	HM	Industriekaufmann, Englisch, Italienisch
Marianne Wagner	MW	Schreibmaschine, Stenografie, Textverarbeitung
Peter Watzmann	PW	PC-Kenntnisse, C-Programmierer
Silke Schmidt	SS	PC-Kenntnisse, Netzwerkkenntnisse
Petra Saubermann	PS	Bürokaufmann ² , kfm. Buchhaltung
Lars Laus	LL	Industriekaufmann, Außendienst Erfahrung
Markus Sputnik	MS	Bürokaufmann, Organisation
Claudia Schön	CS	PC-Kenntnisse, C-Programmierer ³
Willibald Fusel	WF	Einzelhandelskaufmann, Verkaufserfahrung
Harald Hirsch	HH	Verkaufserfahrung

Um die Organigramme nicht zu unübersichtlich zu gestalten, wurden folgende für alle Beispiele geltenden Beziehungen nicht abgebildet:

Klaus Schwarz (KS) hat die Prokura.

Jörg Schwarz (JS) ist Stellverteter von Klaus Schwarz (KS).

Peter Watzmann (PW) ist Stellvertreter von Hilde Weiß (HW).

Silke Schmidt (SS) ist Stellvertreterin von Sebastian Blau (SB).

Die eingezeichneten Kanten stehen für die herkömmliche Weisungsbefugnis.

Beispiel 1:

Das erste Beispiel repräsentiert eine rein linienartig strukturierte Unternehmung, die in Abteilungen gegliedert ist. Das Gliederungskriterium spielt hier keine Rolle; denkbar wäre eine zielorientierte Strukturierung, also eine Gliederung nach Produkten.

1 geschlechtsspezifische Unterscheidungen sind in der Datenbank nicht sinnvoll

2 s.o.

3 s.o.

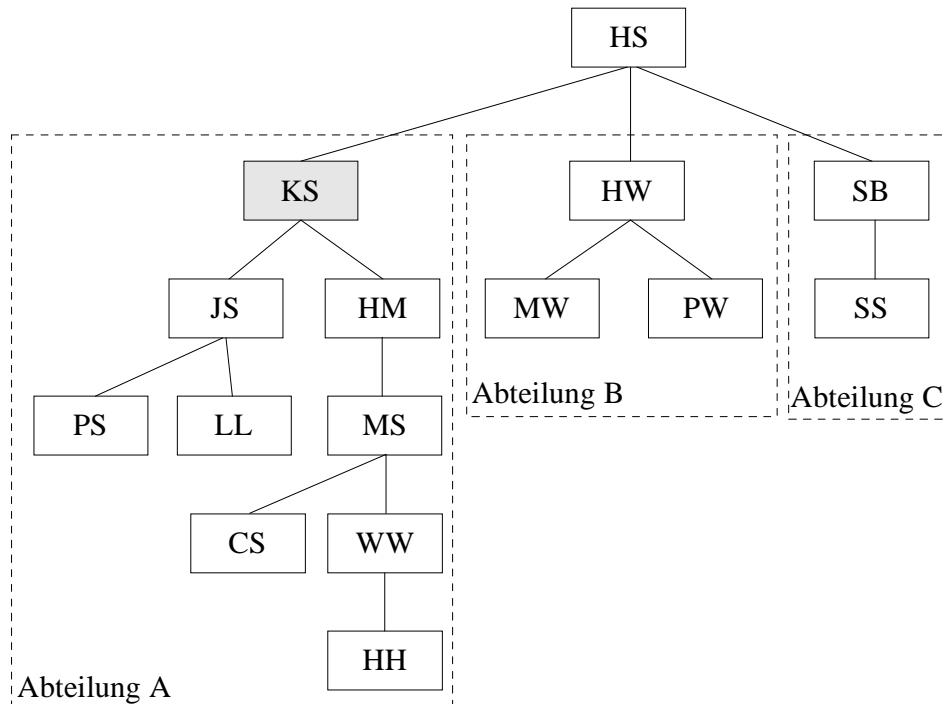


Abbildung 2.2: Beispiel einer linienförmigen Organisationsstruktur

Beispiel 2:

Im zweiten Beispiel wird eine matrixartig strukturierte Unternehmung dargestellt. Die Matrix wird institutionalisiert durch die Entscheidungsdimensionen Ziel und Feld, d.h. die Gliederung erfolgt einerseits nach Produkten und andererseits nach Regionen. Die Mitarbeiter in den "Schnitten" der Bereiche haben im Prinzip jeweils zwei Vorgesetzte. Entscheidungen müssen von beiden Einflußfaktoren gemeinsam getroffen werden.

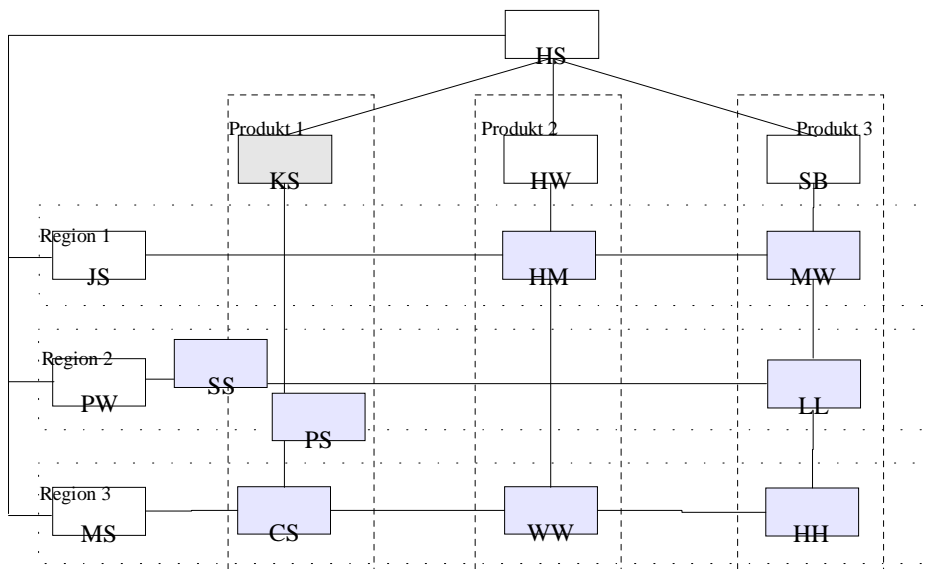


Abbildung 2.3: Beispiel einer matrixartigen Organisationsstruktur

Beispiel 3:

Eine reine Projektorganisation ist Gegenstand des dritten Beispiels. Auffällig bei dieser Organisationsform ist die flache Hierachiestruktur; es gibt vergleichsweise wenig "Vorgesetzte". Für jedes Projekt ist ein Projektleiter bestimmt, die übrigen Projektmitglieder sind einander gleichberechtigt.

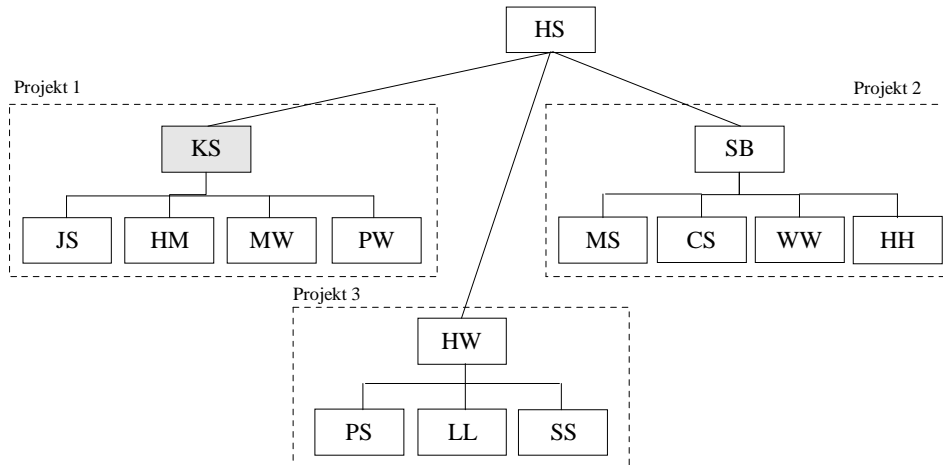


Abbildung 2.4: Beispiel einer projektartigen Organisationsstruktur

Beispiel 4:

Das vierte Beispiel ist eine Mischung aus den drei vorhergehenden Organisationsformen, die jede für sich eine Reinform der jeweiligen Gliederungsart darstellen. Insofern dürfte diese letzte Organisationsstruktur am ehesten der Realität entsprechen. Die zweite Entscheidungsdimension wird in diesem Fall durch das Controlling repräsentiert.

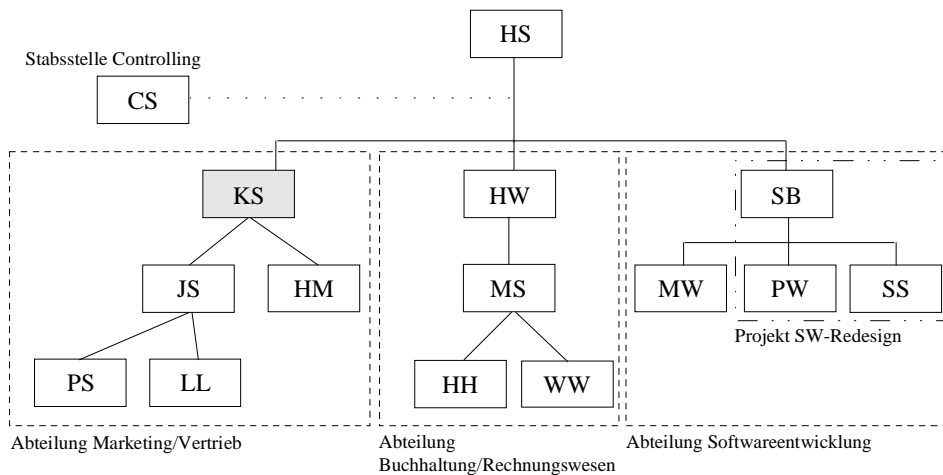


Abbildung 2.5: Beispiel einer gemischten Organisationsstruktur

2.3. Weitere Anforderungen

Betrachtet man den Ablauf einer Interaktion zwischen der Ablaufsteuerungskomponente (Workflow Engine) und der Organisationskomponente, so fällt auf, daß sich dieser nach dem Client-Server-Schema verhält: Die Workflow Engine stellt eine Anfrage, liefert zur

Bearbeitung Parameter und erwartet eine Antwort; eine klassische Request-Response-Interaktion. Das heißt, daß sich diese Art der Realisierung anbietet. Da die einzelnen Komponenten des WfMS aufgrund der zu erwartenden heterogenen DV-Strukturen ohnehin "netzwerkfähig" sein sollten, bedeutet das nur einen geringen Mehraufwand.

Wird aber die Organisationskomponente schon als Server realisiert, so steht einer Benutzung durch andere Anwendungen nichts entgegen. Allerdings ist es möglich, daß diese anderen Anwendungen, z.B. ein Teleconferencing-System, andere Funktionalitäten benötigen als ein WfMS. Daher ist die Schnittstelle zwischen WfMS und Organisationskomponente aufzuteilen in einen WfMS-spezifischen Teil und einen organisationsdatenbankspezifischen Teil. Letzterer ist in seiner Funktionalität mächtiger als der WfMS-Teil, da das WfMS nur eine Untermenge der möglichen Funktionen benötigt.

Möglicherweise benötigt eine andere Anwendung aber auch andere, zusätzliche Daten als die, die im Datenmodell der Organisationskomponente als Teil eines WfMS vorgesehen wurden. In einem solchen Fall muß das Datenmodell in gewissen Grenzen erweiterbar sein, d.h. es müssen zwar Daten hinzugefügt werden können, das Datenmodell selbst darf nicht veränderbar sein.

Schließlich soll die Organisationskomponente so portabel wie möglich realisiert werden.

2.4. Grobentwurf

In diesem Kapitel sollen die bereits gewonnenen Erkenntnisse zur Festlegung eines ersten, groben Systementwurfs verwendet werden.

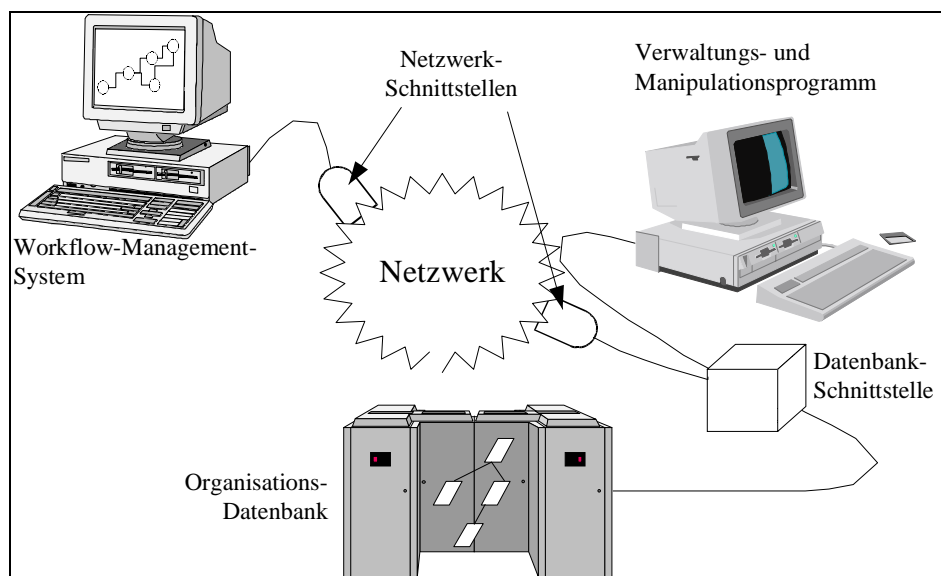


Abbildung 2.6: Modell der Organisationskomponente

2.4.1. Organisationsdatenbank

Die Organisationsdatenbank ist das Kernstück der Organisationskomponente. Sie muß in der Lage sein, die Aufbauorganisation eines beliebigen Unternehmens zu speichern. Dazu gehören einerseits die Mitarbeiter der Unternehmung mit all ihren unternehmensrelevanten Eigenschaften und Fähigkeiten und andererseits die im Unternehmen existierenden

organisatorischen Einheiten (Stellen, Abteilungen, usw.) und ihre organisationsrelevanten Beziehungen zueinander abbildbar sein. Dazu ist ein geeignetes, möglichst kompaktes Datenmodell zu entwerfen.

Als Datenbank wird ein relationales Datenbanksystem gewählt.

Das Datenmodell muß in gewissen Grenzen erweiterbar sein, d.h. es muß möglich sein, die Datenbasis zu ergänzen, ohne das Datenmodell anzutasten.

2.4.2. Verwaltung der Organisationsdatenbank

Um die Organisationsstruktur einer Unternehmung in der Organisationsdatenbank speichern zu können, wird ein Werkzeug benötigt, mit dessen Hilfe die relevanten Daten in die Datenbank eingegeben werden können. Dieses Werkzeug soll eine graphische Benutzerschnittstelle haben, um die Bedienung zu vereinfachen. Bereits eingegebene Organisationsstrukturen sollen graphisch dargestellt werden können.

2.4.3. Anwendungsschnittstelle

Die Wiederverwendbarkeit der Organisationsdatenbank wird durch die Trennung der Datenbankschnittstelle von der WfMS-Schnittstelle möglich. Diese stellt dem WfMS Funktionen zur Verfügung, mit deren Hilfe es seine Anforderung nach einem Bearbeiter übermitteln kann. Innerhalb der WfMS-Schnittstelle werden die Funktionsaufrufe ausgewertet. Diese Funktionsaufrufe enthalten sowohl die Auswahlkriterien, nach denen ein Bearbeiter gesucht werden soll, als auch optionale Parameter, mit denen das WfMS den Suchvorgang beeinflussen kann. Als Ergebnis der Auswertung generiert die Schnittstelle Funktionsaufrufe, die an die Organisationsdatenbankschnittstelle weitergeleitet werden. Erst diese greift auf die eigentliche Datenbank zu. Das Ergebnis einer Suche gelangt auf demselben Weg zum WfMS.

Das Kommunikationsprinzip wird im folgenden Bild verdeutlicht:

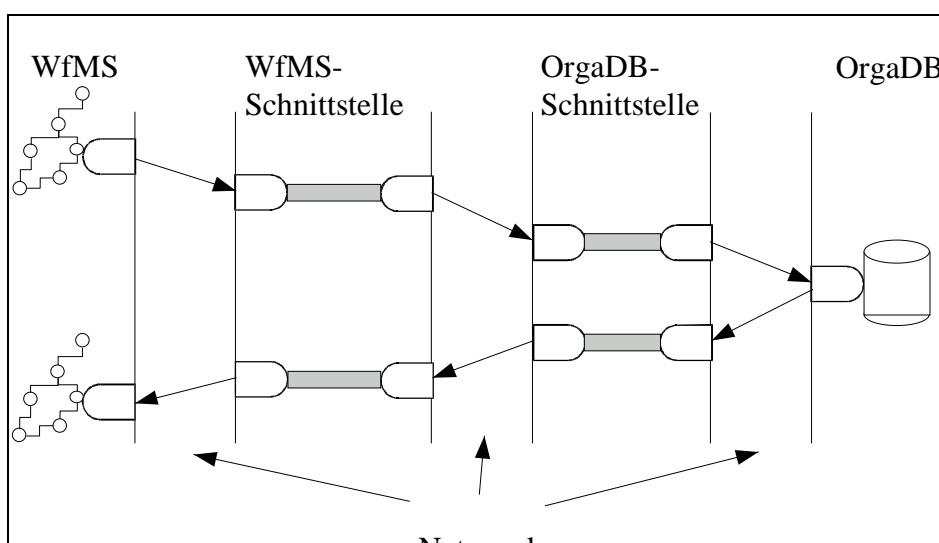


Abbildung 2.7: Kommunikationsprinzip der Anwendungsschnittstelle

3. Entwurf der Organisationsdatenbank

In diesem Kapitel soll das Datenmodell der Organisationsdatenbank festgelegt werden. Hierbei soll den Anforderungen aus Kapitel 2 Rechnung getragen werden.

3.1. Begriffsbildung

In den vorhergehenden Kapiteln wurde die Anwendungsumgebung ausführlich beschrieben und auch Begriffe verwendet, deren Bedeutungen im Anwendungskontext klar sind. Diese Begriffe sollen an dieser Stelle noch einmal allgemein erläutert werden; auf die im Datenmodell verwendete spezielle Bedeutung wird nachfolgend eingegangen.

Folgende Begriffe sind im Anwendungsumfeld der Organisationskomponente von Bedeutung:

- **Organigramm:** Als Organigramm wird eine grafische Darstellung einer Organisation bezeichnet; es enthält die zur Organisation gehörenden *Personen* bzw. *Stellen* und *Gruppen* und ihre *Beziehungen* zueinander.
- **Stelle:** Die Stelle ist die atomare Organisationseinheit in einem Unternehmen. Sie kann eine *Aufgabe* sowie *Eigenschaften* (insbesondere *Kompetenzen* und *Rechte*) beinhalten. Eine Stelle wird i.a. mit einer oder mehreren *Personen* besetzt, kann aber auch (zeitweise) unbesetzt bleiben.
 - **Beziehungen zwischen Stellen:** Stellen stehen in Beziehungen zueinander. Dies können hierarchische (Vorgesetzter-Untergebener-Beziehung) oder nicht-hierarchische Beziehungen (Kollege-Kollege-Beziehung) sein.
- **Gruppe:** Eine Gruppe ist eine Menge von *Stellen*. Eine Gruppe kann auch Untergruppen enthalten.
 - **Projekt:** Menge von *Stellen*, die in einem Projekt zusammengefaßt sind.
 - **Abteilung:** Menge von *Stellen*, die in einer Abteilung zusammengefaßt sind.
 - **Beziehungen zwischen Gruppen:** Alle Beziehungen zwischen Stellen können auch für Gruppen gelten.
- **Person:** Objekt (Ressource) in der Organisation, das *Fähigkeiten* hat und zur Bearbeitung von *Aufgaben* herangezogen werden kann. Personen besetzen Stellen.
 - **Beziehungen zwischen Personen:** Personen haben keine unternehmensrelevanten Beziehungen untereinander.
- **Eigenschaft:** Es kann unterschieden werden zwischen nicht übertragbaren, gewissermaßen persönlichen Eigenschaften, die Personen zugeordnet werden, und Eigenschaften, die eher funktions- bzw. stellenorientiert sind und damit von einer Stelle auf eine andere übertragen werden können.
 - **Kompetenz:** Kompetenz bedeutet die formale Zuständigkeit für einen Vorgang oder

für eine Sache. Kompetenzen sind typischerweise übertragbare Eigenschaften.

- **Recht:** Bestimmte Tätigkeit oder Funktion (z.B. Prokura), die nur an ausgewählte Personen bzw. Stellen vergeben wird.
- **Fähigkeit:** Eigenschaft (Attribut) von Personen, die diese zur Bearbeitung von Aufgaben eignen.
 - **Qualifikation:** Eine Fähigkeit.
 - **Berufstitel, Berufsbezeichnung:** Eine Qualifikation.
 - **Anforderung:** Eine Eigenschaft, die zur Bearbeitung einer Aktivität benötigt wird.
 - **Beziehungen zwischen Eigenschaften:** Eigenschaften können untereinander ähnlich sein (z.B. C-Programmierer, Pascal-Programmierer). In solchen Fällen können mit Hilfe von Oberbegriffen Eigenschaftshierarchien gebildet werden. Dies ist aber nur für Fähigkeiten sinnvoll.
 - **Eigenschaftsgrade:** Personen können Eigenschaften in verschiedenen Graden besitzen (Schulenglisch, Englisch fließend in Wort und Schrift, Englisch als Muttersprache)
- **Beziehungsarten:** In einer Organisation sind verschiedene Beziehungsarten zwischen Organisationseinheiten denkbar. Die typische Beziehungsart ist die Linienhierarchie, also eine Vorgesetzter-Untergebener-Beziehung. Ebenso gibt es aber auch Stellvertreterbeziehungen u.a.

3.2. Das Entity-Relationship-Diagramm

Nachdem die Begriffe, die für den Datenbankentwurf von Bedeutung sind, definiert und erklärt wurden, kann nunmehr das Entity-Relationship-Modell der Organisationsdatenbank entwickelt werden. Zunächst sollen die abzubildenden Entities und Relationships mit ihren Attributen identifiziert und erläutert und später mögliche Varianten vorgestellt werden.

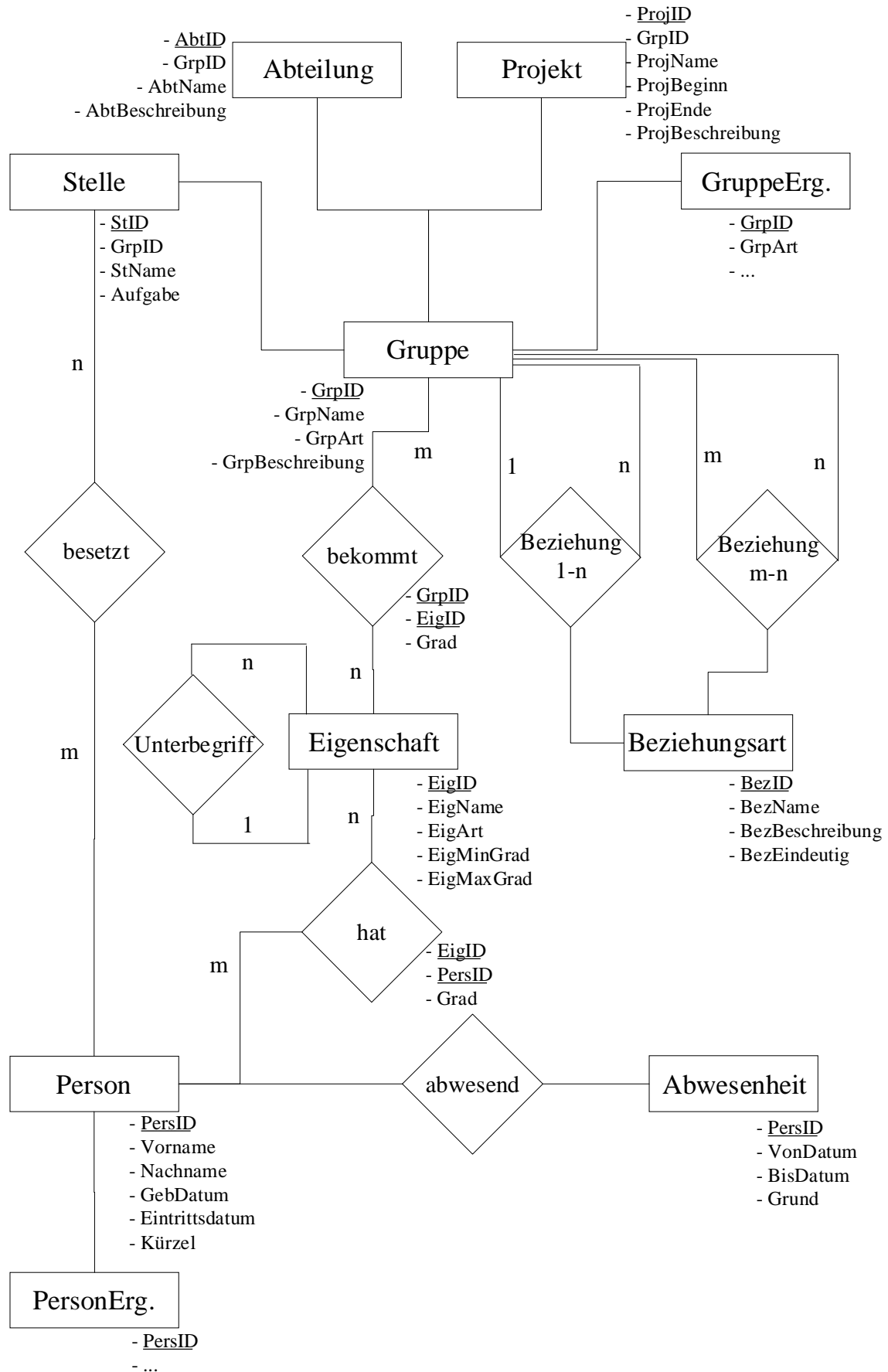


Abbildung 3.1: ER-Diagramm der Organisationsdatenbank

3.2.1. Beschreibung der Entitätsmengen

- **Stelle.**

Eine Organisation setzt sich aus Stellen zusammen. Jede dieser Stellen hat eine Bedeutung im Unternehmen, ihre Aufgabe oder auch Stellenbeschreibung. An eine Stelle können *Eigenschaften*, meist Kompetenzen und Rechte, geknüpft sein; diese können auf andere Stellen (z.B. Stellvertreter) übertragen werden.

Im Datenmodell werden Stellen als spezielle einelementige Gruppen aufgefaßt; die Zuordnung von Eigenschaften erfolgt auf Gruppenebene, da auch Gruppen unternehmensrelevante Eigenschaften haben. Dagegen werden nach wie vor Stellen von Personen besetzt; diese übliche Vorgehensweise soll realitätsnah modelliert werden.
- **Person.**

Die Ressource im Unternehmen, mit der Stellen besetzt werden. Dabei können sich auch mehrere Personen eine Stelle teilen (z.B. Teilzeitarbeitskräfte) oder eine Person kann mehrere Stellen, reale und nicht-reale, innehaben (z.B. Mitarbeit in mehreren Projekten). Personen haben *Eigenschaften*, die sie zur Bearbeitung von Aufgaben eignen können; solche Fähigkeiten können nicht übertragen werden. Auch Kompetenzen können Personen verliehen werden (z.B. Prokura).
- **Gruppe.**

Stellen können zu Gruppen zusammengefaßt werden. Dabei ist das Gruppierungskriterium zunächst einmal zweitrangig; man definiert eine Gruppe und legt dann fest, welche Stelle(n) zu dieser Gruppe gehört(en). Um den Begriff der Gruppe so neutral und abstrakt wie möglich halten zu können, sollen auch einzelne Stellen als einelementige Gruppen verstanden werden. Auf diese Weise können Beziehungen zwischen Gruppen (und damit auch zwischen einzelnen Stellen) definiert werden, beispielsweise eine Weisungshierarchie. Auch das Stellvertreterkonzept ist auf diese Weise realisierbar. *Stelle*, *Abteilung* und *Projekt* sind Spezialisierungen der Gruppe und haben auch spezielle Eigenschaften wie Abteilungsbezeichnung oder Projektname und -dauer.
- **Eigenschaft.**

Eigenschaften können übertragbar oder persönlich sein. Sie gliedern sich im wesentlichen in Fähigkeiten und Kompetenzen. Die Lösung von Aufgaben erfordert Fähigkeiten, *Personen* haben sie. Um eine *Stelle* geeignet zu besetzen, sollten die Anforderungen mit den im Unternehmen vorhandenen Eigenschaften einigermaßen zur Deckung gebracht werden. Um Eigenschaften besser einordnen zu können, wird eine Eigenschaftenhierarchie aufgebaut; dabei werden die Eigenschaften von Stufe zu Stufe weiter spezialisiert. Damit ist es möglich, Ähnlichkeiten zwischen Eigenschaften zu etablieren. Wird beispielsweise eine Eigenschaft nicht gefunden ("Pascal-Programmierer"), so kann der Oberbegriff ("Programmierer") als passend gefunden werden. Desweiteren ist es möglich zu spezifizieren, welche Eigenschaft in welchem Grade zur Bearbeitung einer Aufgabe nötig ist. Auch damit gewinnt man bei der Bestimmung einer "passenden" Person Freiheitsgrade. Aber auch Kompetenzen können "angefordert" werden, für sie gilt aber das Gesagte in gleicher Weise.
- **Beziehungart.**

Gruppen können miteinander in Beziehungen stehen, z.B. Vorgesetzter-Beziehung, Kollege-Beziehung oder Stellvertreter-Beziehung. Um möglichst jede Art von Organisationsstruktur modellieren zu können, ist die *Beziehungsart* frei definierbar.

- **Abwesenheit.**
Um die Verfügbarkeit, oder besser: Nicht-Verfügbarkeit, eines Mitarbeiters modellieren zu können, wird ein zusätzliches Entity benötigt, in dem das entsprechende Zeitintervall und der Grund der Abwesenheit eingetragen wird.

3.2.2. Beschreibung der Relationships

- **Relation besetzt**
Diese Relation beschreibt, welche *Person* welche *Stelle* besetzt. Da eine Person mehrere Stellen besetzen kann, z.B. durch Mitarbeit in mehreren Projekten, und mehrere Personen eine Stelle sich teilen können (Teilzeitstellen), muß diese Beziehung als m-n-Relation modelliert werden.
- **Relation hat**
Diese Relation ordnet *Personen Eigenschaften*, in der Regel Fähigkeiten, zu. Auch diese Beziehung ist als m-n-Relation zu realisieren, da mehrere Personen dieselbe Eigenschaft und Personen mehrere Eigenschaften haben können. Zusätzlich kann angegeben werden, in welchem *Grad* eine Person eine Fähigkeit besitzt.
- **Relation bekommt**
Diese Relation ordnet, analog zur Relation **hat**, *Eigenschaften* zu, allerdings den *Gruppen*. Diese stellenbezogenen Eigenschaften sind meist Kompetenzen und Rechte. Ansonsten gilt das für die vorher beschriebene Relation Gesagte.
- **Relation Unterbegriff**
Um Eigenschaftenhierarchien aufbauen zu können, die Ähnlichkeiten zwischen Eigenschaften etablieren, wird diese Relation eingeführt. Sie ordnet einer *Eigenschaft* eine andere zu, die als Spezialisierung aufgefaßt wird. Damit jede Eigenschaft maximal einen Oberbegriff hat, ist diese Beziehung als 1-n-Relation zu realisieren.
- **Relation Beziehung(m-n)**
Mit Hilfe dieser Relation werden, unter Angabe der jeweiligen *Beziehungsart*, die Beziehungen zwischen *Gruppen* realisiert. Die m-n-Variante dieser Relation erlaubt die entsprechende Modellierung von Beziehungen (Beispiel: Kollege-Kollege-Beziehung).
- **Relation Beziehung(1-n)**
Die 1-n-Variante dieser Relation ermöglicht die Modellierung von 1-n-Beziehungen zwischen *Gruppen*, die es im Unternehmen geben kann (z.B. eine Stelle darf immer nur genau einer Kostenstelle zugeordnet werden).
- **Relation abwesend**
Diese Relation ermöglicht die Erfassung von Fehlzeiten einer Person, um die Verfügbarkeit derselben im Bedarfsfall zur Laufzeit überprüfen zu können.

3.2.3. Ergänzungen und Bemerkungen zum Datenmodell

Das bisher vorgestellte Datenmodell ist ausreichend für die Speicherung von Aufbauorganisationen, da alle organisatorischen Einheiten und ihre Beziehungen zueinander sowie die Zuordnungen von Personen auf die organisatorischen Einheiten abgebildet werden können. Es ist jedoch noch nicht dafür geeignet, die gewünschte Funktionalität im Zusammenhang mit einem WfMS zu erbringen. Hierzu sind einige Ergänzungen zum Datenmodell notwendig.

Gleichgültig, welche Attribute für die Entities Person und Stelle oder Gruppe usw. im Datenmodell vorgesehen werden - sie reichen sicherlich in irgendeiner Unternehmung nicht aus oder sind unzutreffend. Daher ist es erforderlich, daß benutzerdefinierte Attribute unterstützt werden. Dies wird mit der Einführung neuer Tabellen erreicht: Die Entities Person und Gruppe (mit ihren Spezialisierungen Stelle, Abteilung und Projekt) bekommen als je eine (weitere) Spezialisierung eine Tabelle, deren Attributmenge erweitert werden kann. Die Zuordnung der Werte erfolgt über die entsprechenden Schlüssel. Das bedeutet: Es gibt eine Tabelle *PersonErgänzung* bzw. *GruppeErgänzung* mit den vorgegebenen Attributen PERSID bzw. GRPID und GRPART, auf die der Benutzer zugreifen kann, um Attribute hinzuzufügen; das Löschen von Attributen ist nicht möglich. Da Stellen, Abteilungen und Projekte spezielle Gruppen sind, können auch sie in der *GruppeErgänzung*-Tabelle neue Attribute erhalten. Auch ermöglicht diese Art der Modellierung die Definition neuer Spezialisierungen der Entität *Gruppe*; man definiert einen neuen Wert des Attributs GRPART sowie die gewünschten Attribute der neuen Spezialisierung in der *GruppeErgänzung*-Tabelle; allerdings muß die Organisationsdatenbankschnittstelle die entsprechende Funktionalität zur Verwendung solcher "neuer" Gruppenarten zur Verfügung stellen.

Durch die Unterstützung benutzerdefinierter Attribute für Personen und Gruppen kann die Attributanzahl in den eigentlich relevanten Entities soweit begrenzt werden, daß eine Identifizierung eines Datensatzes noch möglich ist. Die Namen der Attribute der einzelnen Entities erklären sich selbst; auf eine ausführliche Besprechung kann daher verzichtet werden.

Um ihre Hauptfunktionalität, Bearbeiter für Aktivitäten anhand von Eigenschaften und Vorgaben zu ermitteln, erfüllen zu können, werden gewisse Beziehungen zwischen Gruppen benötigt; diese Beziehungen sind Teil der Suchalgorithmen und dürfen in der Datenbank daher nicht gelöscht werden. Dazu gehören die *Vorgesetzter*- bzw. *Untergebener*-Beziehung, die *gehört-zu*-Beziehung und die *Stellvertreter*-Beziehung. Auf die Suchalgorithmen selbst soll später eingegangen werden.

3.2.4. Varianten

Da eine Stelle als einelementige Gruppe aufzufassen ist, wäre es möglich, mit der Beziehung *besetzt* die Entities Person und Gruppe zu verknüpfen, d.h. eine Person besetzt eine Gruppe. Diese Betrachtungsweise erscheint aber doch zu weit hergeholt; die atomare Organisationseinheit im Unternehmen ist die Stelle, die sich über eine Teilaufgabe im Unternehmen definiert. Der Begriff der Stelle ist so elementar und auch so gebräuchlich, daß nicht auf ihn verzichtet werden sollte.

Eine weitere Variante des Datenmodells ergibt sich, wenn man die Beziehungen zwischen Gruppen in einer einzigen Relation ablegt. Dies müßte eine m-n-Beziehung sein, da diese

Beziehung im Betrieb vorkommen kann. Damit wäre auch eine 1-n-Beziehung realisierbar; ein entsprechendes Attribut in der *Beziehungsarten*-Tabelle könnte die Programmlogik in der gewünschten Weise steuern. Doch genau darin liegt das Argument für die zweite Relation: Da im Unternehmen 1-n-Beziehungen zwischen Gruppen denkbar sind, muß das Datenmodell diese auch unterstützen; die Programmlogik sollte nicht an die Stelle des Datenmodells treten.

3.3. Das Datenbankschema

Aus der Diskussion ergibt sich das folgende Datenbankschema:

```
CREATE TABLE PERSON (PERSID INTEGER NOT NULL PRIMARY KEY,
  VORNAME VARCHAR(30) NOT NULL,
  NACHNAME VARCHAR(50) NOT NULL,
  GEBDATUM DATE NOT NULL,
  EINTRITTSDATUM DATE,
  KUERZEL VARCHAR(10));

CREATE TABLE BEZIEHUNGSART (BEZID INTEGER NOT NULL PRIMARY KEY,
  BEZNAME VARCHAR(127) UNIQUE NOT NULL,
  BEZBESCHREIBUNG VARCHAR(255),
  BEZEINDEUTIG INTEGER NOT NULL);

CREATE TABLE EIGENSCHAFT (EIGID INTEGER NOT NULL PRIMARY KEY,
  EIGNAME VARCHAR(127) UNIQUE NOT NULL,
  EIGART VARCHAR(10) NOT NULL,
  EIGMINGRAD INTEGER,
  EIGMAXGRAD INTEGER);

CREATE TABLE GRUPPE (GRPID INTEGER NOT NULL PRIMARY KEY,
  GRPNAME VARCHAR(127) UNIQUE NOT NULL,
  GRPART VARCHAR(10) NOT NULL,
  GRPBESCHREIBUNG VARCHAR(255));

CREATE TABLE ABTEILUNG (GRPID INTEGER NOT NULL UNIQUE,
  ABTID INTEGER NOT NULL PRIMARY KEY,
  ABTNAME VARCHAR(127) UNIQUE NOT NULL,
  ABTBESCHREIBUNG VARCHAR(255));

CREATE TABLE PROJEKT (GRPID INTEGER NOT NULL UNIQUE,
  PROJID INTEGER NOT NULL PRIMARY KEY,
  PROJNAME VARCHAR(127) UNIQUE NOT NULL,
  PROJBEGINN DATE,
  PROJENDE DATE,
  PROJBESCH VARCHAR(255));

CREATE TABLE STELLE (GRPID INTEGER NOT NULL UNIQUE,
  STID INTEGER NOT NULL PRIMARY KEY,
  STNAME VARCHAR(127) UNIQUE NOT NULL,
  AUFGABE VARCHAR(255));

CREATE TABLE BEKOMMT (GRPID INTEGER NOT NULL,
  EIGID INTEGER NOT NULL,
  GRAD INTEGER,
  PRIMARY KEY (GRPID, EIGID),
  FOREIGN KEY (GRPID) REFERENCES GRUPPE(GRPID),
  FOREIGN KEY (EIGID) REFERENCES EIGENSCHAFT(EIGID));
```

3. Entwurf der Organisationsdatenbank

```
CREATE TABLE BESETZT (PERSID INTEGER NOT NULL,
    STID INTEGER NOT NULL,
    PRIMARY KEY (PERSID, STID),
    FOREIGN KEY (PERSID) REFERENCES PERSON(PERSID),
    FOREIGN KEY (STID) REFERENCES STELLE(STID));

CREATE TABLE BEZIEHUNG_1N (GRPID1 INTEGER NOT NULL,
    GRPID2 INTEGER NOT NULL,
    BEZID INTEGER NOT NULL,
    PRIMARY KEY(GRPID1, GRPID2, BEZID),
    FOREIGN KEY (GRPID1) REFERENCES GRUPPE(GRPID),
    FOREIGN KEY (GRPID2) REFERENCES GRUPPE(GRPID),
    FOREIGN KEY (BEZID) REFERENCES BEZIEHUNGSART(BEZID));

CREATE TABLE BEZIEHUNG_MN (GRPID1 INTEGER NOT NULL,
    GRPID2 INTEGER NOT NULL,
    BEZID INTEGER NOT NULL),
    PRIMARY KEY(GRPID1, GRPID2, BEZID),
    FOREIGN KEY (GRPID1) REFERENCES GRUPPE(GRPID),
    FOREIGN KEY (GRPID2) REFERENCES GRUPPE(GRPID),
    FOREIGN KEY (BEZID) REFERENCES BEZIEHUNGSART(BEZID));

CREATE TABLE HAT (PERSID INTEGER NOT NULL,
    EIGID INTEGER NOT NULL,
    GRAD INTEGER,
    PRIMARY KEY (PERSID, EIGID),
    FOREIGN KEY (PERSID) REFERENCES PERSON(PERSID),
    FOREIGN KEY (EIGID) REFERENCES EIGENSCHAFT(EIGID));

CREATE TABLE UNTERBEGRIFF (EIGID1 INTEGER NOT NULL,
    EIGID2 INTEGER NOT NULL,
    PRIMARY KEY (EIGID1, EIGID2),
    FOREIGN KEY (EIGID1) REFERENCES EIGENSCHAFT(EIGID),
    FOREIGN KEY (EIGID2) REFERENCES EIGENSCHAFT(EIGID));

CREATE TABLE ABWESEND (PERSID INTEGER NOT NULL,
    VONDATUM DATE NOT NULL,
    BISDATUM DATE NOT NULL,
    GRUND VARCHAR(127),
    PRIMARY KEY (PERSID, VONDATUM, BISDATUM),
    FOREIGN KEY (PERSID) REFERENCES PERSON(PERSID));

CREATE TABLE PERSONERGAENZUNG (PERSID INTEGER NOT NULL);

CREATE TABLE GRUPPEERGAENZUNG (GRPID INTEGER NOT NULL,
    GRPART VARCHAR(10) NOT NULL);
```

3.4. Beispiele

An dieser Stelle sollen die Beispiele aus Kapitel 2.2.2. mit Hilfe des vorgestellten Datenmodells in einer relationalen Datenbank gespeichert werden. Da zumindest das vierte Beispiel, die gemischte Organisationsform, einigermaßen realitätsnah sein dürfte, kann das Datenmodell als erprobt gelten, wenn die Speicherung dieser (und der anderen) fiktiven Unternehmenstrukturen möglich ist.

3.5. Vergleich mit einem anderen Modell

Das Software-Labor der Universität Stuttgart hat im Rahmen des Projektbereichs Workflow Management ein Objekt- (und Beziehungs-)Metamodell entwickelt, das als "Typdefinition für eine Klasse von Modellsystemen" [SINZ96] betrachtet werden kann, aus der alternative Modellsysteminstanzen abgeleitet werden können.

Dieses Objekt-Metamodell besteht aus den drei Blöcken Prozeß-, Ressource- und Qualifikations- und Kompetenzstruktur. Abbildung 3.2 zeigt eine vereinfachte Darstellung, in der aus Übersichtlichkeitsgründen nur die Beziehungen der Subobjekte innerhalb der Blöcke, nicht aber die zwischen den Blöcken dargestellt wurden. Um die ableitbaren Modellinstanzen flexibel zu halten, wurden die meisten Beziehungen als m-n-Beziehungen realisiert. Die rekursiven Beziehungen ermöglichen Hierarchiebildungen der jeweiligen Subobjekte. Aus Platzgründen muß auf die Angabe von Attributen weitgehend verzichtet werden.

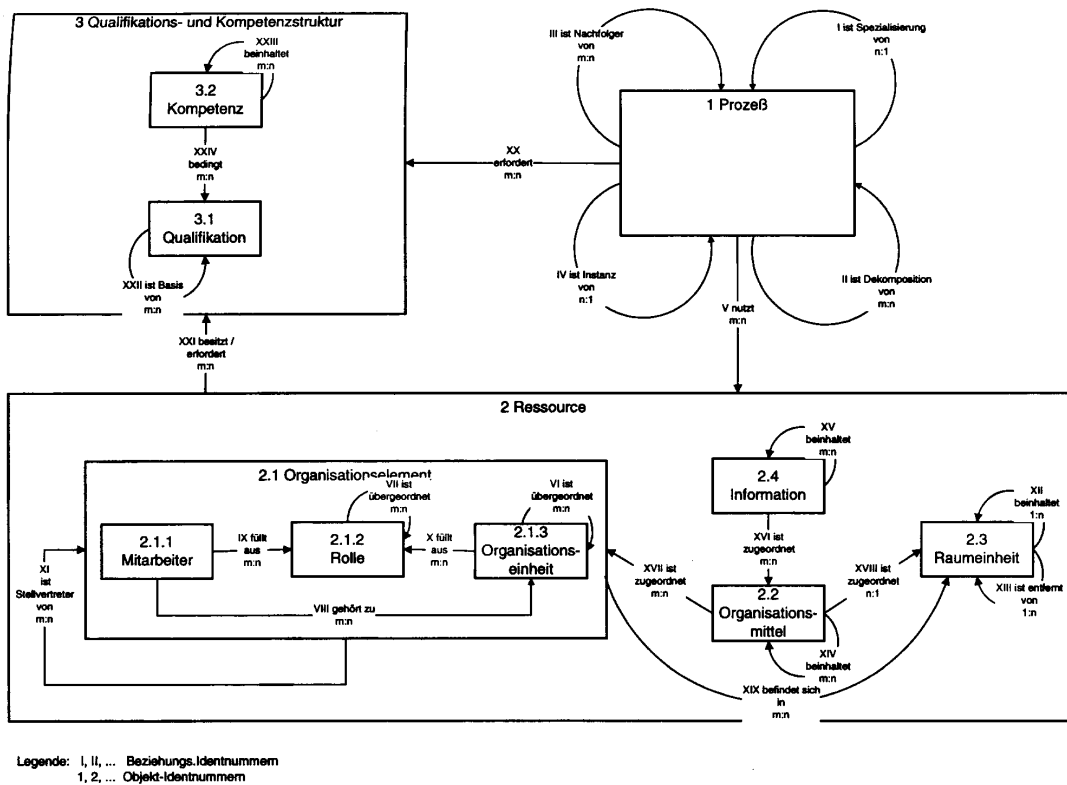


Abbildung 3.2: Metamodell [HEI96. S.159]

Der Block Prozeß ermöglicht die Modellierung der Beziehungen zwischen Aufbau- und Ablauforganisation. Eine Instanz des Objekts Prozeß kann beliebig detailliert sein; sowohl ein in groben Zügen modellierter Kernprozeß als auch ein einzelner Prozeßschritt sind denkbar. Die rekursiven Beziehungen dienen der Modellierung der Spezialisierung (I), der Dekomposition (II) mit höherem Detaillierungsgrad, von Vorgänger-Nachfolger-Beziehungen (III) und der Instanzenbildung (IV) für die operative Steuerungsebene.

Der Block Ressource ist in die Subobjekte Organisationselement, Organisationsmittel, Raumeinheit und Information untergliedert. Das Subobjekt Organisationselement ist weiter unterteilt in Mitarbeiter, Rolle und Organisationseinheit. Das zentrale Konzept ist hier die Rolle: Es werden Rollenhierarchien gebildet (VII), die von Organisationseinheiten (X) oder von Mitarbeitern (IX) ausgefüllt werden. Mitarbeiter werden Organisationseinheiten zugeordnet (VIII). Die Stellvertreterbeziehung ist dem Organisationselement insgesamt zugeordnet (XI); damit sind Stellvertreterregelungen zwischen Mitarbeiter, Rollen und Organisationseinheiten modellierbar. Das Subobjekt Organisationsmittel enthält alle Sachmittel einschließlich Hard- und Software. Eine Dekomposition von Organisationsmitteln wird durch die rekursive Beziehung XIV ermöglicht; weiterhin werden Organisationsmittel sowohl Organisationselementen, also Mitarbeiter, Rolle oder Organisationseinheiten, zugeordnet (XVII) als auch Raumeinheiten (XVIII). Das Subobjekt Raumeinheit ist ebenfalls hierarchisch darstellbar (XII). Da auch im Zeitalter elektronischer Medien Entfernungen relevant sein können, sind diese abbildbar (XIII). Organisationselemente können Raumeinheiten zugeordnet werden (XIX).

Das Subobjekt Information beschreibt den hierarchischen Aufbau der verwendeten Informationen (XV) und ordnet diese Organisationsmitteln zu (XVI). Der Block Qualifikations- und Kompetenzstruktur enthält die Subobjekte Kompetenz und Qualifikation; beide sind hierarchisch beschreibbar (XXII bzw. XXIII). Komponenten bzw. Qualifikationen, die einem Organisationselement zugewiesen werden (XXI), müssen

"zusammenpassen", d.h. die Beziehung XXIV muß im jeweiligen Zusammenhang sinnvoll sein. Die Übereinstimmung von Kompetenz und Verantwortung ist nicht explizit modellierbar; es wird angenommen, daß diese Frage implizit im Subobjekt Kompetenz geklärt ist. Eine Instanz dieses Metamodells wurde von der Abteilung VII (Allgemeine Betriebswirtschaftslehre und Wirtschaftsinformatik) des Betriebswirtschaftlichen Instituts der Universität Stuttgart mit Hilfe der Datenbank Microsoft Access implementiert. Das Datenmodell dieser Implementierung zeigt Abbildung 3.3.

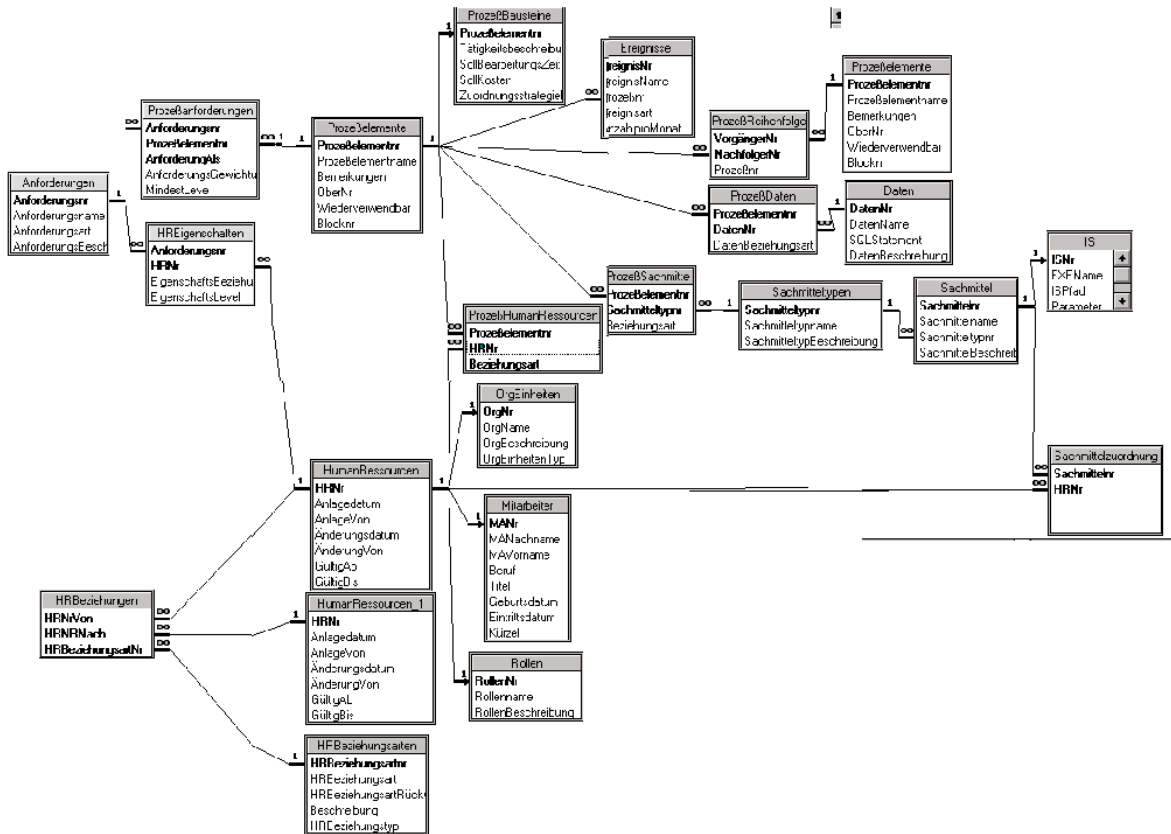


Abbildung 3.3: Datenmodell der MS-Access-Implementierung des Metamodells

Zunächst ist festzuhalten, daß es sich bei der hier abgebildeten Implementierung um eine integrierte Lösung handelt, d.h. Prozeß- und Ressourcenkontrolle werden in einem gemeinsamen Datenmodell abgebildet. Die Schnittstellen zwischen den beiden Bereichen liegen in den Tabellen *HREigenschaften*, die eine Verbindung zwischen den Anforderungen eines Prozesses und den Eigenschaften (= Fähigkeiten) einer sog. Human Ressource herstellt, und *ProzeßHumanRessourcen*, die direkt die Tabellen *Prozeßelemente* und *HumanRessourcen* verbindet. Eine "Human Ressource" kann dabei eine Organisationseinheit, ein Mitarbeiter oder eine Rolle sein. Welche Bedeutung eine Rolle hat, geht aus dem Modell nicht eindeutig hervor.

Interessant für einen Vergleich mit der in dieser Arbeit entworfenen Organisationskomponente ist in etwa die untere Hälfte der Abbildung, in der sich die organisationsrelevanten Modellierungen finden. Auffällig ist zunächst, daß auch in diesem Modell die Beziehungen nicht fest vorgegeben sind; sie können, im Gegensatz zum Metamodell (Abbildung 3.3), frei gewählt werden. Dagegen ist die Organisationsstruktur des Metamodells originalgetreu umgesetzt worden, mit Ausnahme der Beziehungen. Die Unterscheidung zwischen 1-n- und m-n-Beziehungen wird durch ein Attribut in der Be-

ziehungsarten-Tabelle realisiert; dies erfordert zusätzliche Programmlogik der Anwendung. Bei der Suche nach Eigenschaften (im Modell *Anforderungen* genannt) ist das Modell auf reines "Pattern-Matching" angewiesen, allerdings wird zwischen Eigenschaften, Qualifikation, Kompetenz usw. unterschieden. Eine Ähnlichkeit zwischen Eigenschaften ist nicht implementiert.

Zusammenfassend ist zu sagen, daß das in dieser Arbeit vorgestellte Modell einer Organisationskomponente offener sowohl in Bezug auf eine Verwendung außerhalb eines WfMS-Kontextes als auch bezüglich Erweiterungen seitens des Benutzers ist. Durch den Aufbau einer Eigenschaftenhierarchie zusätzlich zu einer Qualifizierung von Eigenschaften ist das hier entwickelte Modell auch flexibler bezüglich der Suche nach Bearbeitern.

4. Die Organisationsdatenbankverwaltung

Als Teil eines WfMS besteht die Aufgabe der Organisationskomponente darin, eine oder mehrere Personen als Bearbeiter einer Aktivität anhand der von ihr geforderten Eigenschaften aus der Menge der im Unternehmen beschäftigten Mitarbeiter zu ermitteln. Die dazu erforderlichen Informationen werden in der Organisationsdatenbank abgelegt. Zur Eingabe und Pflege des Datenbestandes ist ein Werkzeug erforderlich; dieses Programm soll in diesem Kapitel entworfen werden.

4.1. Anforderungen an das Programm

Der Datenbestand der Organisationsdatenbank unterteilt sich in die Bereiche Stammdaten und Verknüpfungen. Unter den Stammdaten sollen die Datensätze in den Entity-Tabellen verstanden werden; in den Relationship-Tabellen werden die Stammdaten in Beziehung zueinander gebracht.

Das Verwaltungsprogramm muß sowohl die Eingabe von Stammdaten als auch die Herstellen der relevanten Beziehungen unterstützen. Dazu müssen die entsprechenden Auswahl- und Eingabemasken zur Verfügung gestellt werden.

Zur Vereinfachung der Bedienung des Programms soll zusätzlich eine grafische Oberfläche zur Verfügung stehen.

4.2. Anforderungen an den Benutzer

Die Organisationsdatenbank enthält eine Repräsentation der Aufbauorganisation eines Unternehmens. Es versteht sich von selbst, daß der Benutzer des Programms zur Verwaltung dieser Datenbank - und damit der in ihr gespeicherten Informationen - profunde Kenntnisse über die Organisationsstruktur des Unternehmens besitzen sollte. Er muß jede Strukturänderung sofort in die Datenbank übertragen; dazu gehören auch Fehlzeiten von Mitarbeitern, die durch Krankheit, Urlaub oder Dienstreisen verursacht werden, sowie Einstellungen und Entlassungen von Mitarbeitern. Umstrukturierungen müssen selbstverständlich ebenso sofort in der Datenbank abgebildet werden. Das bedeutet, daß dieser Benutzer aufgrund der Informationsvielfalt, die ihm zur Verfügung stehen muß, eine zentrale Stellung im Unternehmen einnimmt. Eine Verzögerung bei der Umsetzung neuer Strukturen ist nicht hinnehmbar; das WfMS ist auf die Gültigkeit der in der Organisationsdatenbank vorhandenen Informationen angewiesen.

4.3. Entwurf des Verwaltungsprogramms

Die Grundfunktionalität eines Datenbankverwaltungsprogramms besteht im Hinzufügen, Ändern und Löschen von Daten in der Datenbank. Das bedeutet, daß sowohl Eingabe- als auch Anzeigeelemente zur Verfügung gestellt werden müssen.

In diesem Abschnitt sollen die relevanten Daten und Funktionen modelliert werden, die für das Verwaltungsprogramm von Bedeutung sind. Die Gegenständlichkeit der relevanten Daten legen einen objektorientierten Entwurf nahe.

Ein Ergebnis des Entwurfs werden Anforderungen an die Funktionalität der Organisationsdatenbankschnittstelle sein.

4.3.1. Modellierung der Entities

Die Entities werden als Klassen modelliert; ihre Attribute werden als Instanzvariablen aufgefaßt. Ein Objekt dieser Klasse repräsentiert einen Datensatz der entsprechenden Tabelle. Jedes Objekt dieser Klassen besitzt die Fähigkeit, "sich zu löschen", d.h. den Datensatz, den sie repräsentieren, aus der Datenbank zu entfernen; ebenso sind die weiteren Grundfunktionalitäten "Ändern" und "Hinzufügen" in jeder Klasse präsent.

4.3.2. Modellierung der Relationships

Die Relationships enthalten in der Regel lediglich Identifikatoren der an ihnen teilnehmenden Entities; es bietet sich daher an, sie als zusätzliche Methoden der entsprechenden Entity-Klassen aufzufassen. Beispielsweise hat die Klasse PERSON eine Methode *besetztStelle(StID)*, die die entsprechenden Identifikatoren in die Tabelle *besetzt* einträgt; die Klasse STELLE hat als Gegenstück eine Methode *besetztDurch*, die die Person(en) liefert, die mit dieser Stelle verknüpft ist (sind).

4.4. Modellierung der Datenbank

Da die Organisationskomponente möglicherweise in einem heterogenen Umfeld betrieben wird, könnte sich die Organisationsdatenbank auf einem entfernten Rechner befinden. Um dies vor dem Anwender zu verbergen, wird die Datenbank als eigene Klasse aufgefaßt. Diese Klasse enthält alle vom Verwaltungsprogramm benötigten Funktionen (Abfragen, Hinzufügen, Ändern und Löschen für jede Tabelle); diese werden als Methoden aufgefaßt. Als Klassenvariable, d.h. als Variable, die systemweit nur einen Wert hat und die sich alle Objekte dieser Klasse teilen, wird die Verbindung zur physischen Datenbank modelliert. Dies stellt sicher, daß alle Objekte, die auf die Datenbank zugreifen müssen, sich "lokal" ein Objekt der Klasse Organisationsdatenbank erzeugen können und trotzdem alle gemeinsam auf nur eine physische Datenbank zugreifen.

4.5. Bedienung des Verwaltungsprogramms

Das Verwaltungsprogramm gliedert sich in einen maskenorientierten und einen grafischen Teil. Die Stammdaten werden in Bildschirmmasken erfaßt; die Beziehungen können entweder Datensatz für Datensatz in Masken oder mit allen Teilnehmern an einer Beziehung insgesamt in einer grafischen Darstellung bearbeitet werden.

4.5.1. Bedienung der Maskenoberfläche

Die Benutzung der Bildschirmmasken verläuft stets dreistufig: In der Hauptmaske wird die zu bearbeitende Tabelle, getrennt nach Stammdaten und Beziehungen, ausgewählt, nachdem eine Verbindung zur Datenbank aufgebaut wurde. Im nächsten Schritt ist die Aktion, die durchgeführt werden soll (Hinzufügen, Ändern, Löschen), anzugeben. Schließlich gelangt der Benutzer zu der entsprechenden Eingabe- oder Auswahlmaske, in die er die zutreffenden Daten einträgt bzw. ändert. Wird die Aktion "Löschen" gewählt, so ist ein Ändern der angezeigten Daten nicht sinnvoll und daher auch nicht möglich.

4.5.2. Bedienung der grafischen Oberfläche

Wählt der Benutzer eine Beziehung zur Bearbeitung aus, so hat er die Wahl zwischen der oben beschriebenen Maskenbearbeitung, in der immer genau ein Datensatz bearbeitet werden kann, oder einer grafischen Darstellung aller an der Beziehung teilnehmenden Gruppen. Da eine solche Beziehung nicht notwendigerweise als Baum organisiert ist, muß die Darstellung als gerichteter Graph erfolgen, in dem die Knoten die an der Beziehung teilnehmenden Gruppen als Objekte und die Kanten als Beziehung interpretiert werden.

Der Benutzer markiert einen Knoten und hat dann die Auswahl zwischen den Aktionen Hinzufügen, Ändern und Löschen; da das System Beziehung, Beziehungsart und durch die Markierung auch das Objekt zu diesem Zeitpunkt kennt, kann die entsprechende Aktion ausgeführt werden. Die Aktion "Ändern" hat nur in Beziehungen Sinn, deren Tabellen nicht nur die Identifikatoren der teilnehmenden Entities enthalten.

Die grafische Oberfläche greift nur auf die Beziehungstabellen zu; ein Löschen von Stammdaten ist nicht möglich. Der Grund für diese Beschränkung liegt in der Gefahr von Fehlbedienungen: Wird ein Objekt versehentlich gelöscht, so verschwindet es auch aus allen Beziehungen, Stellen, usw. (referentielle Integrität der Datenbank). Der Aufwand für die Restaurierung einer solchen Fehlbedienung rechtfertigt den Komfort dieser Funktionalität nicht.

4.6. Zusammenfassung

Der Entwurf des Verwaltungsprogramms für die Organisationsdatenbank impliziert ein Schichtenmodell: Die Objekte einer Anwendung greifen über eine (Netzwerk-)Schnittstelle auf die Datenbank zu (Abbildung 4.1).

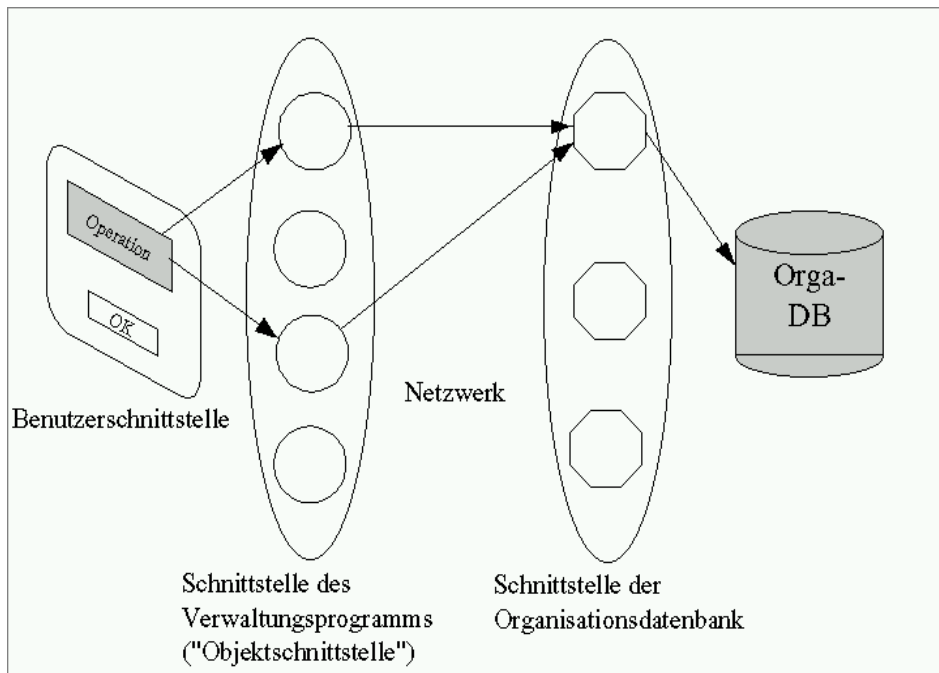


Abbildung 4.1: Schema des Verwaltungsprogramms

5. Die Anwendungsschnittstelle

In diesem Kapitel soll die Schnittstelle zwischen dem WfMS (als Anwender) und der Organisationskomponente entworfen werden. Dazu ist zunächst zu klären, welche Anfragen seitens des WfMS möglich und sinnvoll sind. Dann ist der Mechanismus zu entwerfen, mit dem das WfMS und die Organisationskomponente miteinander kommunizieren; da das Umfeld möglicherweise in einem heterogenen Netzwerk besteht, ist dieses Problem nicht trivial. Schließlich ist zu klären, welche funktionalen Anforderungen die Anwendungsschnittstelle an die Datenbankschnittstelle stellt und wie die beiden Schnittstellen ihrerseits miteinander kommunizieren.

5.1. Anforderungen an die Anwendungsschnittstelle

Die eigentliche Aufgabe der Anwendungsschnittstelle besteht in der Weiterleitung eines WfMS-Auftrags an die Organisationsdatenbank. Da die Organisationsdatenbank auch von anderen Anwendungen angesprochen werden kann, wird die Schnittstelle aufgeteilt in einen WfMS-spezifischen Teil und in einen Organisationsdatenbankspezifischen Teil. Mit "Anwendungsschnittstelle" ist der WfMS-spezifische Teil gemeint.

Der Auftrag, der an die Organisationsdatenbank weitergeleitet werden soll, besteht in einem Suchauftrag nach einem oder mehreren Bearbeitern für eine bestimmte Aktivität. Für diese oder diesen Bearbeiter gelten gewisse Suchkriterien; diese sollen nun näher untersucht werden.

5.1.1. Spezifikation eines Bearbeiters

Die Festlegung der Spezifikation, also der Beschreibung, eines Bearbeiters ist das zentrale Problem in der Anwendungsschnittstelle, da diese Spezifikation die Formulierung der Suchkriterien festlegt, durch deren Angabe die Suche erst möglich wird.

Ein Bearbeiter kann nach folgenden Methoden gesucht werden:

- Man spricht ihn direkt an (mit der *Personalnummer*). Diese Art der Spezifikation ist nur in Verbindung mit anderen Spezifikationsmethoden sinnvoll; da man den Bearbeiter kennt, braucht man ihn nicht zu suchen
- Man spezifiziert (eine Menge von) *Eigenschaften*; der gesuchte Bearbeiter ist der, auf den diese Anforderungen am besten passen.
- Man gibt eine *Beziehung* und eine *Gruppe* an; der gesuchte Bearbeiter gehört zu der resultierenden Gruppe, so daß evtl. weiter ausgewählt werden muß.
- Man gibt eine *Stelle* an (Name oder Identifikator); der gesuchte Bearbeiter ist derjenige, der sie besetzt. Diese Methode erlaubt die (zumindest indirekte) Spezifikation eines Bearbeiters über die Aufgabe.

Da die Möglichkeit besteht, daß das WfMS nicht nur einen, sondern mehrere Bearbeiter sucht (es kann Aktivitäten geben, die mehrere Bearbeiter erfordern), muß diese Möglichkeit von der Organisationskomponente und damit auch von der Anwendungsschnittstelle

unterstützt werden. Das bedeutet einerseits, daß das WfMS angeben muß, wieviele Bearbeiter es sucht, und andererseits, daß die oben vorgestellten Suchmethoden nicht ausreichen. Außerdem lassen sie die Möglichkeit außer acht, daß der gefundene Bearbeiter gar nicht verfügbar sein könnte. Weiterhin ist es möglich, daß kein Mitarbeiter die gewünschten Eigenschaften aufweist.

Bereits im Datenbankentwurf wurden diese Möglichkeiten berücksichtigt: Mit der Stellvertreter-Beziehung kann Ersatz für einen nicht verfügbaren Mitarbeiter gefunden werden, und mit einer Eigenschaftenhierarchie kann eine zunächst erfolglose Suche mit ähnlichen Eigenschaften, genauer: mit Oberbegriffen der ursprünglichen Anforderungen, fortgesetzt werden.

Diese zusätzlichen Verfahren bedürfen jedoch der Kontrolle durch das WfMS, d.h. das WfMS entscheidet, ob diese zur Anwendung kommen. Diese Entscheidung wird der Organisationskomponente durch die Angabe von Optionen mitgeteilt; auf diese Weise ist dem WfMS eine beschränkte Einflußnahme auf den Suchprozeß möglich.

5.1.2. Bezug auf vorhergehende Anfragen

Eine weitere Möglichkeit der Beeinflussung des Suchprozesses besteht in der Bezugnahme auf vorhergehende Anfragen an die Organisationskomponente. So könnte das WfMS spezifizieren, daß es für eine bestimmte Aktivität denselben Bearbeiter möchte wie bei der letzten Anfrage für diese Aktivität. Dies impliziert zweierlei: Erstens benötigt die Organisationskomponente ein "Gedächtnis", d.h. einen Log, in dem es den Bearbeiter der letzten Anfrage nachschlagen kann. Da dies eine anwendungsspezifische Anforderung ist, sollte dieser Log in die Anwendungsschnittstelle verlegt werden.

Zweitens muß das WfMS einen Identifikator für die Aktivität angeben; wird eine Aktivität im Rahmen einer Redefinition eines Prozesses verändert, so muß sie einen neuen Identifikator bekommen, da sonst die Zuordnung einer Anfrage auf eine frühere nicht möglich ist.

Der Log ist, entsprechend seiner Funktionalität, einfach aufgebaut: Er muß sich den Aktivitäts-Identifikator und das von der Organisationsdatenbank gelieferte Anfrageergebnis, also eine Personenmenge, merken.

5.1.3. Übersicht über die Suchfunktionalität

Nach diesen Vorüberlegungen können die möglichen Angaben für die Spezifizierung eines Suchauftrag angegeben werden:

- Elementaralgorithmen
 - Personalnummer (direkter Zugriff)
- Menge von Eigenschaften (eine oder mehrere Eigenschaften)
 - Beziehungsart und Gruppe
 - Stelle (Stellenidentifikator oder Stellenname)
- Anzahl der gesuchten Bearbeiter, Default: alle gefundenen

- ONE (Option: Genau ein Bearbeiter gesucht)
- (MORE[, COUNT]) (Option: Mehrere Bearbeiter gesucht; COUNT: Anzahl der gesuchten Bearbeiter)
- Stellvertretung, Default: Stellvertretung zugelassen. Wenn die ausgewählte Person nicht verfügbar ist, wird ohne Rücksicht auf die Entsprechung der Anforderungen der Stellvertreter, so definiert, zurückgeliefert.
 - NODEPUTY (Stellvertretung nicht zugelassen)
 - NEXTBEST (wenn Bearbeiter nicht verfügbar, dann Bearbeiter mit der nächstbesten "Qualifikation", und nicht automatisch der Stellvertreter)
- Eigenschaftenoberbegriffe, Default: zugelassen, wenn zuwenig Bearbeiter gefunden wurden
 - NOSUPERIORATTRIBUTE (nicht zugelassen)
- Bezug auf vorhergehende Auftrag, Default: kein Bezug auf einen vorhergehenden Auftrag
 - SAME (derselbe Bearbeiter wie beim letzten erfolgreich bearbeiteten Auftrag mit dieser Aktivität)
 - OTHER (ein anderer Bearbeiter als beim letzten erfolgreich bearbeiteten Auftrag mit dieser Aktivität)

5.1.4. Erfolglose Suche

Trotz aller Kriterien und Optionen, die die Suche nach einem Bearbeiter in der Organisationsdatenbank spezifizieren, kann eine Suche erfolglos verlaufen; die Eigenschaft, nach der gesucht wird, ist im Unternehmen nicht (mehr) vorhanden, der Bearbeiter ist nicht verfügbar und weder ein Stellvertreter noch ein potentieller Bearbeiter mit ähnlichen Qualifikationen ist vorhanden, usw. In einem solchen Fall kann die Organisations-komponente nur einen entsprechenden Fehler an das WfMS zurückliefern. Dazu müssen die Fehlermeldungen geeignet spezifiziert werden. Wie bzw. ob das WfMS auf einen Fehler reagiert, ist nicht Sache der Anwendungsschnittstelle; für diese ist jeder Auftrag ein neuer Auftrag. Ein "fehlgeschlagener" Auftrag wird nicht im Log vermerkt.

Folgende Fehler können auftreten:

- Verwaltungsprogramm
 - RECORD EXISTS (Datensatz bereits vorhanden, Einfügen nicht möglich)
- Anwendungsschnittstelle
 - ATTRIBUTES NOT AVAILABLE (spezifizierte(n) Eigenschaft(en) sind nicht verfügbar)
 - NO PERSON AVAILABLE (keine geeignete Person verfügbar)

- JOB NOT ASSIGNED (Stelle nicht besetzt)
- NO SUPERIOR GROUP (keine übergeordnete Gruppe definiert)
- NO GROUPS ASSIGNED (keine Untergruppen zugeordnet)
- NO DEPUTY DEFINED (kein Stellvertreter definiert)
- PERSON NOT AVAILABLE (Person nicht verfügbar)

5.2. Formale Spezifikation der Suchfunktion

Die Anwendungsschnittstelle wird vom WfMS im Prinzip durch den Aufruf der Suchfunktion `FINDPERSON` angesprochen. Diese Funktion wird folgendermaßen spezifiziert:

<code>FindPerson(<AktivitätsID>, <Auswahlkriterien>)</code>	
<code><AktivitätsID> ::=</code>	<code>LongInteger</code>
<code><Auswahlkriterien> ::=</code>	<code>(<Einzelkriteriumsliste> [, <Optionsliste>])</code>
<code><Einzelkriteriumsliste> ::=</code>	<code>(<Einzelkriterium> [(, <Einzelkriterium>)*])</code>
<code><Einzelkriterium> ::=</code>	<code>([NOT] (<Direktbezug> <Eigenschaftsbezug> <Beziehungsbezug> <Stellenbezug>)</code>
<code><Direktbezug> ::=</code>	<code>(DIRECT(<Personalnummer>))</code>
<code><Eigenschaftsbezug> ::=</code>	<code>(ATTRIBUTE((<Eigenschaftsname>, <Eigenschaftsgrad> [, (<Eigenschaftsname>, <Eigenschaftsgrad>)*])</code>
<code><Beziehungsbezug> ::=</code>	<code>(RELATIONSHIP(<Beziehungsartname>, <GruppenSpezifikation>)</code>
<code><Stellenbezug> ::=</code>	<code>(POSITION(<Stellenname>))</code>
<code><GruppenSpezifikation> ::=</code>	<code>(GROUP(<Einzelkriteriumsliste>))</code>
<code><Optionsliste> ::=</code>	<code>(OPTION(<Option> [(, <Option>)*])</code>

Über die `<GruppenSpezifikation>` können die verschiedenen Spezifikationsarten miteinander kombiniert werden. Dabei werden die Gruppen betrachtet, zu denen die Bearbeiter, die in `<Einzelkriterium>` referenziert werden, gehören.

Da ein Suchkriterium für einen Bearbeiter auch der Ausschluß z.B. einer Eigenschaft sein kann, muß die Spezifikation auch die Formulierung eines solchen "negativen" Suchkriteriums zulassen. Durch die Verwendung des Schlüsselworts `NOT` ist diese Möglichkeit gegeben.

Die Angabe einer Liste von Suchkriterien (`<Einzelkriteriumsliste>`) impliziert, daß die

gesuchte(n) Person(en) alle Anforderungen erfüllen soll(en), d.h. da jedes Kriterium eine Menge von Bearbeitern referenziert, wird die Auswahl aus der Schnittmenge dieser Personenmengen getroffen.

5.3. Vorgehensweise der Anwendungsschnittstelle

5.3.1. Grundalgorithmus

Wird die Anwendungsschnittstelle aufgerufen, so loggt sie zunächst den Aktivitätsidentifikator. Dann wertet sie die Parameter aus und ruft ihrerseits die entsprechenden von der Organisationsdatenbankschnittstelle zur Verfügung gestellten Funktionen auf. Die konkreten Anforderungen an die Funktionalität der Organisationsdatenbankschnittstelle sollen später erörtert werden.

5.3.2. Auswertung der Parameter

Entsprechend der Spezifikation der Suchfunktion muß der Parameterstring geparkt werden. Dabei werden gleichartige Suchkriterien zusammengefaßt, z.B. alle Eigenschaftsanforderungen werden gemeinsam bearbeitet. Dies ist möglich, da den Suchkriterien eine UND-Semantik zugrundeliegt, d.h. die Angabe von mehreren Suchkriterien impliziert, daß der/die gesuchte(n) Bearbeiter (möglichst) alle Kriterien gleichzeitig erfüllen muß. Das bedeutet, daß die Ermittlung der endgültigen Bearbeitermenge durch Schnittbildung der einzelnen erhaltenen Bearbeitermengen erfolgen kann.

Dieser Auswahlalgorithmus ist sehr einfach; die Realisierung eines anderen jedoch sehr komplex. Soll z.B. aus einer Gruppe von in Frage kommenden Bearbeitern immer abwechselnd, entsprechend einer gewissen Reihenfolge, ein Bearbeiter ausgewählt werden, so ist dafür die Angabe dieser Reihenfolge erforderlich. Diese setzt aber voraus, daß sich die Zusammensetzung dieser Gruppe nicht ändert. Die Einhaltung dieser Vorbedingung kann nicht garantiert werden. Eine andere Möglichkeit (in diesem konkreten Fall) wäre die Abwechslung "nach dem Alphabet"; kommt aber ein Bearbeiter in dieser Gruppe hinzu, so ist ebenfalls die Reihenfolge unterbrochen und damit sinnlos. Andererseits ist die Möglichkeit, einen solchen, alternativen Algorithmus angeben zu können, sicher wünschenswert, wird aber im vorliegenden Entwurf nicht unterstützt.

Eine Ausnahme bei der Zusammenfassung gleichartiger Kriteriumstypen bildet die Gruppenspezifikation. Diese erlaubt die Schachtelung von Suchkriterien und betrachtet die resultierende Bearbeitermenge als Gruppe, auf die wiederum andere Kriterien angewendet werden können. Die Auswertung dieser geschachtelten Suchkriterien muß einzeln "von innen nach außen" erfolgen.

Die Angabe von Optionen beeinflusst die Suche nach Bearbeitern (dazu sind sie da). Die Angabe der Option SAME veranlaßt die Anwendungsschnittstelle, im Log nach dem letzten Bearbeiter für die angegebene Aktivität zu suchen; dabei werden weitere Suchkriterien im Parameterstring ignoriert. Es wird geprüft, ob dieser Bearbeiter aktuell verfügbar ist; wenn ja, so wird dieser Bearbeiter als Ergebnis zurückgeliefert, ansonsten wird die entsprechende Exception (PERSON NOT AVAILABLE) generiert. Wird die Option OTHER angegeben, so wird wiederum im Log nach dem letzten Bearbeiter gesucht und eine zusätzliche

Bedingung mit der Negation dieses Bearbeiters erzeugt (NOT(<Bearbeiter>)).

5.4. Kommunikation

Die Anwendungsschnittstelle als Mittler zwischen dem WfMS und der Organisationsdatenbank soll als Netzwerkdienst entworfen werden, um netzwerkweit, unabhängig vom Standort des Benutzers (dem WfMS), angesprochen werden zu können. Dafür bietet sich ein Entwurf nach dem CORBA-Standard an.

5.4.1. CORBA

Die *Object Management Group (OMG)* ist ein internationaler, nicht profitorientierter Zusammenschluß von Hard- und Softwareherstellern, die die Entwicklung von Technologien für die Produktion verteilter Software zu beeinflussen sucht. Mit der *Object Management Architecture (OMA)* hat die OMG eine Softwarearchitektur spezifiziert, die das Zusammenarbeiten von Anwendungen verschiedener Hersteller ermöglicht, unabhängig davon, für welches Betriebs- oder Hardwaresystem in welcher Programmiersprache sie entwickelt wurden.

Kern dieser Architektur ist der *Object Request Broker (ORB)* als universelles Kommunikationsmedium für beliebig geartete Objekte in verteilten heterogenen Systemen. Der zugehörige Standard heißt *Common ORB Architecture (CORBA)* und zeichnet sich durch folgende Eigenschaften aus:

- **Objektorientierung.** Die grundlegenden Einheiten der Architektur sind Objekte, wobei ein Objekt im Sinne der OMA eine beliebige, eindeutig identifizierbare Einheit ist, also nicht notwendigerweise ein Objekt im Sinne einer Programmiersprache.
- **Verteilungstransparenz.** CORBA-Programme greifen auf entfernte Objekte mit denselben Mechanismen zu wie auf lokale Objekte. Der Aufenthaltsort eines Objekts bleibt für den Klienten in der Regel unbekannt.
- **Effizienz.** Die Architektur für den ORB ist bewußt so gehalten, daß effiziente Implementierungen möglich sind, die z.B. im Falle rein lokaler Kommunikation dem traditionellen Funktionsaufruf nur unwesentlich nachstehen.
- **Hardware-, Betriebssystem- und Programmiersprachenunabhängigkeit.**
- **Offenheit.** Über den ORB können Programme verschiedener Hersteller zusammenarbeiten. Der Anwender erhält dadurch die Freiheit, jede Komponente von dem Hersteller beziehen zu können, der seinen individuellen Bedürfnissen am ehesten gerecht wird. Softwareentwickler erhalten die Chance, die Angebote großer Firmen mit eigenen, spezialisierten Produkten zu ergänzen.

Vergleicht man eine verteilte, OMA-basierte Anwendung mit einem menschlichen Organismus, dann entspricht der ORB dem Nervensystem: Wie die Nervenbahnen alle Gebiete unseres Körpers durchziehen, so verbindet der ORB alle Objekte einer Anwendung miteinander und ermöglicht ihre Kommunikation und damit ihr koordiniertes Zusammenwirken.

Ein Client schickt einen Request (Auftrag) über den ORB an ein Objekt (Server) und erhält

die Antwort wieder über den ORB. Das Objekt bearbeitet einen Request, indem es eine Operation ausführt, die die vom Client mitgelieferten Werte der Argumente verarbeitet, der Zustand des Objekts verändert und Ergebniswerte erzeugt werden. Ein spezielles Ergebnis ist die Meldung einer Programmausnahme (Exception), um dem Client das Auftreten eines Fehlers bei der Bearbeitung eines Requests anzuzeigen.

Ein Client ist eine Softwareinheit, die eine Operation eines Objekts aufrufen möchte. Der Request ist die Nachricht, mit der der Client das Objekt zur Ausführung der Operation auffordert. Die Objekt-Implementierung ist eine Menge von Programmcode und beschreibt das Verhalten eines Objekts. Ein Objekt wird im Rechner durch eine Objekt-Implementierung, zusammen mit den zu diesem konkreten Objekt-Exemplar gehörenden (privaten) Daten repräsentiert.

Erhält der ORB vom Client einen Request, dann ist der ORB für die Lokalisierung der Objekt-Implementierung im Netz, der Übertragen der Daten zum Zielrechner und für die Zustellung des Requests an die Objekt-Implementierung verantwortlich. Verwenden Sender und Empfänger des Requests verschiedene lokale Darstellungsformate für ihre Daten, so übernimmt der ORB deren Konvertierung. Die Übermittlung der Response geschieht analog.

5.4.2. IDL (Interface Definition Language)

Die Sprache IDL stellt u.a. Mittel zur Beschreibung von Interfaces bereit. Diese Beschreibung erfolgt auf konzeptionellem Niveau, da keine Programmiersprache direkt etwas mit IDL-Definitionen anfangen kann. Diese müssen erst in äquivalente Konstrukte der betreffenden Programmiersprache übersetzt werden, so daß diese mit ihren Mitteln (Methoden- oder Funktionsaufrufe usw.) den Zugriff auf ein Interface durchführen kann. Wie den Konstrukten von IDL äquivalente Konstrukte einer Programmiersprache zugeordnet werden, wird in dem für jede Programmiersprache spezifischen *Language-Mapping*-Dokument beschrieben.

5.5. IDL-Spezifikation der Anwendungsschnittstelle

Die Vereinfachung der Parameterauswertung (Kapitel 5.3.2.) ermöglicht auch eine Vereinfachung der Objektschnittstelle. Dieses Interface soll in Anlehnung⁴ der IDL-Schreibweise spezifiziert werden:

```
interface wfms_orga
{
// Zunächst die Fehler, die auftreten können
//
exception RECORD_EXISTS
    { const string meldung = "Datensatz existiert"; };
exception ATTRIBUTES_NOT_AVAILABLE
    { const string meldung = "Eigenschaft(en) nicht vorhanden"; };
exception NO_PERSON_AVAILABLE
    { const string meldung = "Kein Bearbeiter gefunden"; };
exception JOB_NOT_ASSIGNED
    { const string meldung = "Stelle nicht besetzt"; };
exception NO_SUPERIOR_GROUP
```

⁴ Syntax aus Platzgründen vereinfacht: Bei den Parametern der Operationen fehlen die Datentypen

```
        { const string meldung = "Keine übergeordnete Gruppe
                                definiert"; };
exception NO_GROUPS_ASSIGNED
        { const string meldung = "Keine Untergruppen zugeordnet"; };
exception NO_DEPUTY_DEFINED
        { const string meldung = "Kein Stellvertreter definiert"; };
exception PERSON_NOT_AVAILABLE
        { const string meldung = "Person nicht verfügbar"; };
//
// Nun die Variable und Operation der Schnittstelle
//
readonly attribute short op_status;           // Statusmeldung der
                                              // letzten Operation
short findPerson(in suchString, out personenFeld) raises
        (ATTRIBUTES_NOT_AVAILABLE, NO_PERSON_AVAILABLE,
         JOB_NOT_ASSIGNED, NO_SUPERIOR_GROUP, NO_GROUPS_ASSIGNED,
         NO_DEPUTY_DEFINED, PERSON_NOT_AVAILABLE);
}
```

Die Variable `suchString` enthält die Suchkriterien im in Kapitel 5.2. spezifizierten Format.

5.6. Anforderungen an die Datenbankschnittstelle

Die Anwendungsschnittstelle "übersetzt" die Parameter, die ihr vom WfMS geliefert wurden, in Aufrufe der Organisationsdatenbankschnittstelle. Diese sollte die Funktionalität der Anwendungsschnittstelle unterstützen: Sie sollte Funktionen zur Verfügung stellen, auf die die Grundfunktionen der Anwendungsschnittstelle abgebildet werden können:

- Ermitteln einer Gruppe von Personen aus einer Menge von Stellen
- Ermitteln einer Gruppe von Personen aus der Angabe einer Menge von Eigenschaften
- Verfügbarkeit einer Person prüfen
- Ermitteln einer Gruppe von Personen aus der Angabe einer Menge aus Beziehungsart-Gruppe-Paaren
- Ermitteln einer Gruppe von Personen aus der Angabe einer Menge von Gruppe

Zur Realisierung der Algorithmen werden folgende Hilfsfunktionen benötigt, die auch von anderen Anwendungen genutzt werden können:

- Ermitteln eines Oberbegriffs einer Eigenschaft
- Ermitteln der Untergruppen einer Gruppe (über Beziehung gehört-zu)
- Ermitteln der Obergruppen einer Gruppe (über die Beziehung gehört-zu)
- Ermitteln der Stellvertreter einer Person (über die Beziehung ist-Stellvertreter-von)

Weiterhin sind Funktionen erforderlich, die der Anwendungsschnittstelle eine Behandlung der optionalen Parameter zur Beeinflussung der Bearbeitersuche ermöglichen. Im wesentlichen handelt es sich dabei um Funktionen, die aus einer Gruppe von Personen nach gegebenen Kriterien selektieren:

- Auswahl der Person aus einer Personengruppe, die die Anforderungen am besten erfüllt
- Eine Gruppe nach der Güte ordnen, in der sie die Anforderungen erfüllen (der Beste)

zuerst)

Schließlich soll die Datenbankschnittstelle geeignete Fehlermeldungen generieren, um dem Benutzer bzw. der Anwendung die Möglichkeit zu geben, angemessen zu reagieren.

6. Die Schnittstelle der Organisationsdatenbank

Die Schnittstelle der Organisationsdatenbank (oder kurz Datenbankschnittstelle) wird sowohl vom Verwaltungsprogramm als auch von der Anwendungsschnittstelle zum Zugriff auf die Organisationsdatenbank verwendet. In diesem Kapitel sollen diese Schnittstelle entworfen sowie ihre Funktionalität und den dazu gehörigen Algorithmen festgelegt werden.

6.1. Anforderungen an die Datenbankschnittstelle

Die Funktionalität der Datenbankschnittstelle wird im wesentlichen von den Anforderungen der Benutzer bestimmt. Da die Organisationsdatenbank auch von anderen Anwendungen als dem WfMS benutzt werden kann, sollen einige zusätzliche, wünschenswerte Funktionen angeboten werden.

6.1.1. Anforderungen des Verwaltungsprogramms

Die Funktionalität des Verwaltungsprogramms besteht in der Pflege des in der Datenbank vorhanden Datenbestands. Dazu sind die Grundfunktionen Hinzufügen, Ändern und Löschen von Daten notwendig. Diese Funktionen für jede vom Verwaltungsprogramm ansprechbare Tabelle muß die Schnittstelle der Organisationsdatenbank zur Verfügung stellen.

Desweiteren sind aber auch Hilfsfunktionen notwendig, um z.B. auf eine Eingabe eine entsprechende Auswahl zu ermöglichen. Soll beispielsweise eine Gruppe aus einer Beziehung gelöscht werden, so wählt der Benutzer zunächst die Beziehungsart; das System füllt nun die Bildelemente (Auswahllisten) mit den entsprechenden Gruppen, so daß dem Benutzer gleich die korrekten Daten zur Verfügung stehen.

6.1.2. Anforderungen der Anwendungsschnittstelle

Die Anforderungen der Anwendungsschnittstelle zum WfMS an die Funktionalität der Organisationsdatenbankschnittstelle wurden bereits in Kapitel 5.5. formuliert.

6.2. Funktionalität der Datenbankschnittstelle

Die Schnittstelle der Organisationsdatenbank bietet ihren Benutzern somit folgende Funktionalität:

- Hinzufügen, Ändern und Löschen von Datensätzen für jede Tabelle
- Ermitteln einer Personengruppe durch die Angabe von
 - Eigenschaft(en)

- Gruppe(n)
- Stelle(n)
- Beziehungsart und Gruppe
- Hilfsalgorithmen
 - Ermitteln eines Oberbegriffs einer Eigenschaft
 - Ermitteln der Untergruppen einer Gruppe (über Beziehung gehört-zu)
 - Ermitteln der Obergruppen einer Gruppe (über die Beziehung gehört-zu)
 - Ermitteln der Stellvertreter einer Person (über die Beziehung ist-Stellvertreter-von)
 - Prüfung der Verfügbarkeit einer Person
 - Ordnen einer Personengruppe nach der Güte der Anforderungserfüllung (Rangfolge)

6.3. Algorithmen

In diesem Kapitel sollen die Algorithmen entworfen werden, die die Funktionalität der Datenbankschnittstelle realisieren sollen. Diese Algorithmen werden in einer Pseudo-Code-Schreibweise notiert.

6.3.1. Grundfunktionen

Die Grundfunktionalität des Verwaltungsprogramms kann direkt in der Datenbanksprache formuliert werden, da es sich dabei um reine Datenmanipulationen (Hinzufügen, Ändern und Löschen) handelt. Auch die Auswahlelemente (Listboxen) der Auswahlmasken des Verwaltungsprogramms werden durch entsprechende (direkte) Abfragen der Datenbank bestückt.

Eine Ausnahme bilden die Tabellen *Person* und *Gruppe*. Da das Datenmodell für diese Entitäten die Angabe benutzerdefinierter Attribute zulässt, müssen zusätzlich die entsprechenden Ergänzungstabellen in die Datenbankzugriffe einbezogen werden.

Die Elementarfunktionen der Organisationsdatenbankschnittstelle sind komplexer, da sie die Gegebenheiten des Datenbestands zu berücksichtigen haben und die Anfragen der Anwendung nicht trivial sind.

Ermitteln einer Bearbeitergruppe durch Angabe einer Stelle

Bei diesem Algorithmus wird die Stelle durch ihren Identifikator repräsentiert.

```
ErmittlePerson(Stelle)
{
for each Person aus (Person, Stelle) í besetzt-Relation do
    if (Person verfügbar) then
        Person zur Ergebnismenge hinzufügen
    else if not(NODEPUTY) then
```

```
        ErmittleStellvertreter(Person)
    end if
end for
return Ergebnismenge
}
```

Ermitteln einer Bearbeitergruppe durch Angabe von Eigenschaften

Dieser Algorithmus besteht aus zwei Teilen: Der Algorithmus `ErmittlePerson(Eigenschaft)` bestimmt zunächst die Personenergebnismenge aus der Angabe einer Eigenschaft ohne den optionalen Grad der Fähigkeit bzw. Qualifikation

```
ErmittlePerson(Eigenschaft)
{
if Eigenschaft ist persönliche Eigenschaft then
    for each Person aus (Person, Eigenschaft) Í hat-Relation do
        if (Person verfügbar) then
            Person zur Ergebnismenge hinzufügen
        else if not (NODEPUTY) and not (NEXTBEST) then
            ErmittleStellvertreter(Person)
        end if
    end for
else
    for each Gruppe aus (Gruppe, Eigenschaft) Í bekommt-Relation
        (ErmittlePerson(Gruppe))
    end for
end if
if (zuwenig Bearbeiter gefunden) and not(NOSUPERIORATTRIBUTE) then
    ErmittleOberbegriff(Eigenschaft)
    ErmittlePerson(Oberbegriff(Eigenschaft))
end if
return Ergebnismenge
}
```

Wird der Grad der Fähigkeit bzw. Qualifikation zusätzlich angegeben, d.h. ist dieser Grad für die Bearbeitersuche relevant, so muß dieser Grad abgeprüft werden, bevor eine Person zur Ergebnismenge hinzugefügt wird.

Der zweite Teil des Algorithmus, `ErmittleSchnittmenge`, bildet die Schnittmenge der erhaltenen Personenergebnismengen, d.h. er ermittelt die Personen, die in allen Ergebnismengen vertreten sind, und bildet daraus eine neue Ergebnismenge. Diese Ergebnismenge enthält also alle Personen, die alle geforderten Eigenschaften (ggf. in den geforderten Graden) besitzen.

Ermittlung einer Bearbeitergruppe durch Angabe von Beziehungsarten und Gruppen

Dieser Algorithmus sucht, analog zum vorigen, zunächst eine Bearbeitergruppe durch die Angabe eines einzelnen Beziehungsart-Gruppe-Paars.

```
ErmittlePerson(Beziehungsart, Gruppe)
{
if (Beziehungsart eindeutig) then
    for each Gruppe_1 aus (Gruppe, Beziehungsart, Gruppe_1) Í
```

```

                                Beziehung_1n-Relation do
                                ErmittlePerson(Gruppe)
                                end for
else
    for each Gruppe_1 aus (Gruppe, Beziehungsart, Gruppe_1) Í
        Beziehung_1n-Relation do
            ErmittlePerson(Gruppe)
        end for
end if
return Ergebnismenge
}
```

Um die Bearbeitung mehrerer Beziehungsart-Gruppen-Paare zu ermöglichen, wird wieder der Algorithmus `ErmittleSchnittmenge` verwendet, um diejenigen Personen zu finden, die in allen Gruppen enthalten sind.

Ermitteln von Bearbeitergruppen durch Angabe von Gruppen

Dieser Algorithmus arbeitet rekursiv.

```

ErmittlePerson(Gruppe)
{
if (Gruppe = Stelle) then
    ErmittlePerson(Stelle)
    Hinzufügen von Person zur Ergebnismenge
else
    ErmittleUntergruppen(Gruppe)
    for each Untergruppe(Gruppe) do
        ErmittlePerson(Untergruppe(Gruppe))
    end for
end if
return Ergebnismenge
}
```

6.3.2. Hilfsfunktionen

Die folgenden Algorithmen werden zur Realisierung der Grundfunktionen benötigt; sie können aber auch von Anwendungen (WfMS oder andere) ausgeführt werden.

Ermitteln eines Oberbegriffs einer Eigenschaft

Dieser Algorithmus ermittelt den Oberbegriff einer Eigenschaft; da diese Beziehung als 1-n-Beziehung definiert ist, gibt es jeweils nur einen Oberbegriff zu jeder Eigenschaft.

```

ErmittleOberbegriff(Eigenschaft)
{
if (Oberbegriff, Eigenschaft) Í Unterbegriff-Relation then
    return Oberbegriff
else
    return Fehler
end if
}
```

Ermitteln der Untergruppenmenge einer Gruppe

Dieser Algorithmus ermittelt die direkten Untergruppen einer Gruppe, d.h. diejenigen Gruppen, die in einer Gruppe enthalten sind, er verwendet dazu die Beziehung *gehört-zu*, die sowohl als 1-n- als auch als m-n-Beziehung definiert ist.

```
ErmittleUntergruppen(Gruppe)
{
for each Untergruppe aus (Untergruppe "gehört-zu-mehrdeutig" Gruppe)
    Í gehört-zu-mehrdeutig-Relation do
    Hinzufügen von Untergruppe zur Ergebnismenge
end for
for each Untergruppe aus (Untergruppe "gehört-zu-eindeutig" Gruppe)
    Í gehört-zu-eindeutig-Relation do
    Hinzufügen von Untergruppe zur Ergebnismenge
end for
return Ergebnismenge
}
```

Ermitteln der Obergruppen einer Gruppe

Dieser Algorithmus verwendet dieselben Beziehungsarten wie der vorhergehende; es werden lediglich die beiden Gruppen-Identifikatoren in den Beziehungs-Relationen vertauscht.

Ermitteln der Stellvertreter einer Person

```
ErmittleStellvertreter(Person)
{
for each Stelle aus (Person, Stelle) Í besetzt-Relation do
    Stelle « Gruppe
    for each Stellvertretergruppe aus (Stellvertretergruppe "ist-
        Stellvertreter-von" Gruppe Í Beziehungs-mn-
        Relation do
        ErmittlePerson(Stellvertretergruppe)
        Hinzufügen der Personen zur Ergebnismenge
    end for
end for
return Ergebnismenge
}
```

Prüfung der momentanen Verfügbarkeit einer Person

Dieser Algorithmus prüft, ob eine Person zum aktuellen Zeitpunkt verfügbar ist; dies genügt im Zusammenhang mit der Anwendung WfMS, da die Zuordnung Aktivität-Bearbeiter zur aktuellen Laufzeit erfolgt. Sollte die Dauer einer Aktivität ebenfalls ein Kriterium sein, so muß anstelle des aktuellen Zeitpunkts das entsprechende Zeitintervall abgeprüft werden.

```
TesteVerfügbarkeit(Person)
{
for each VonDatum, BisDatum aus (Person, VonDatum, BisDatum) Í
```

```
                                abwesend-Relation do
    if aktuelles Datum in (VonDatum, BisDatum) then
        return "nicht verfügbar"
    end if
end for
return "verfügbar"
}
```

Ordnen einer Personengruppe nach der Rangfolge ihrer Anforderungserfüllung

Um im Falle einer Stellvertretung diejenige Person auswählen zu können, die die gestellten Anforderung am nächstenbesten erfüllt, ist es notwendig, eine Rangfolge aller ermittelten Personen aufzustellen. Dies kann in Erweiterung der Funktion `ErmittleSchnittmenge` geschehen, die im Rahmen der Grundfunktionen besprochen wurde. Dazu wird jeder Person die Anzahl der erfüllten Anforderungen zugeordnet. Eine Prioritätenvergabe ist nicht vorgesehen; der Algorithmus ist jedoch entsprechend erweiterbar.

6.3.3. Beurteilung der Algorithmen

Viele der vorgestellten Algorithmen beinhalten mehrfache Datenbankzugriffe. Da diese über das Netzwerk erfolgen, muß mit einer entsprechend geringen Ausführungsgeschwindigkeit gerechnet werden. Für die Anwendung WfMS ist dies kein primäres Problem, da es sich nicht um eine interaktive Anwendung handelt. Für das Verwaltungsprogramm (und evtl. andere interaktive Anwendungen) spielt dieser Umstand eine große Rolle: Eine "zu langsame" Anwendung wird nicht gern eingesetzt.

Einige Algorithmen benutzen explizit Beziehungen. Diese müssen in der Datenbank fixiert sein, d.h. sie dürfen von Benutzern nicht gelöscht werden.

6.4. Kommunikation mit den Benutzern

Das möglicherweise heterogene Umfeld der Organisationskomponente, und damit auch der Schnittstelle der Organisationsdatenbank, legt ein Entwurf auch dieser Schnittstelle als CORBA-Objekt nahe.

6.4.1. IDL-Spezifikation der Datenbankschnittstelle

Analog zur abgekürzten Schreibweise der IDL-Spezifikation der Anwendungsschnittstelle soll hier die Schnittstelle der Organisationsdatenbank spezifiziert werden. Aus Platzgründen sind die Elementaroperationen des Verwaltungsprogramms nicht enthalten.

```
interface orga_db
{
    readonly attribute short op_status;
    short ermittlePerson(in stellenMenge, out personenMenge);
    short ermittlePerson(in eigenschaftenMenge, out personenMenge);
}
```

6. Die Schnittstelle der Organisationsdatenbank

```
short ermittlePerson(in beziehungsArtMenge, in gruppenMenge, out
    personenMenge);
short ermittlePerson(in gruppenMenge, out personenMenge);
short ermittleOberbegriff(in eigenschaft, out oberbegriff);
short ermittleUntergruppen(in gruppe, out gruppenMenge);
short ermittleObergruppen(in gruppe, out gruppe);
short ermittleStellvertreter(in person, out personenMenge);
boolean testeVerfuegbarkeit(in person);
short ordnePersonenMenge(in personenMenge, in eigenschaftenMenge, out
    personenMenge);
short takeBest(in personenMenge, in eigenschaftenMenge, out
    personenMenge);
}
```

7. Realisierung

In diesem Kapitel sollen die Mittel zur Realisierung, also das Zielsystem, die verwendete Programmiersprache und das verwendete Datenbanksystem vorgestellt werden. Ferner soll die Implementierung besprochen werden.

7.1. Mittel zur Realisierung

Die Organisationskomponente soll mit Hilfe der Programmiersprache Java realisiert werden. Als relationales Datenbanksystem soll IBM DB2 verwendet werden.

7.1.1. Die Programmiersprache Java

Java wurde 1991 von der Firma Sun Microsystems im Rahmen eines Software-Forschungsprojektes (Codename Oak) für elektronische Haushaltsgeräte wie Fernseher, Toaster usw. entwickelt. Das Ziel der Entwicklung war eine Sprache, die klein und schnell ist und die leicht auf die verschiedensten Hardware-Plattformen portiert werden kann. Diese Eigenschaften machen Java zu einer idealen Sprache für Anwendungen, die über das Internet, also einer extrem heterogenen Netzwerkumgebung, auf beliebige Zielrechner übertragen werden sollen.

Prinzipiell sind zwei Arten von Java-Anwendungen möglich: *Applets* laufen innerhalb eines sog. Browsers ab, der die Java-Laufzeitumgebung enthält; *Applications* dagegen sind eigenständige Programme, die keinen Browser benötigen.

Die Java-Syntax weist große Ähnlichkeit zu der von C++ auf; die Programmiersprache enthält aber nur einen Teil des Sprachumfangs von C++. Da die Sprache gegenwärtig noch erweitert wird, sollen an dieser Stelle nur einige Merkmale von Java genannt werden:

Java ist plattformunabhängig

Unter der Plattformunabhängigkeit einer Programmiersprache versteht man die Eigenschaft, in ihr geschriebene Programme auf verschiedenen Rechnersystemen (z.B. IBM RS/6000, Sun Sparc, DEC Alpha und PC) ablaufen lassen zu können; der Aufwand für eine Portierung ist dabei minimal zu halten. Auf der Quelltextebene ist Plattform-unabhängigkeit bereits (mehr oder weniger) Realität; Voraussetzung dafür ist ein Compiler für die Programmiersprache, der in dieser Sprache geschriebene Programme in die jeweilige Maschinensprache übersetzen kann. Auf diese Weise kann ein Quelltext für mehrere Plattformen genügen. Ein Beispiel für eine solche Programmiersprache ist C++.

Java ist eine Interpretersprache, d.h. der Java-Compiler erzeugt einen Bytecode, der zur Laufzeit von einem Interpreter interpretiert und ausgeführt wird. Dieser Bytecode ist plattformunabhängig und daher auf jeder Hardwareplattform lauffähig, für die ein Interpreter verfügbar ist.

Ein solcher Interpreter ist in allen javafähigen WWW-Browsern (Netscape Navigator, Microsoft Internet Explorer, HotJava usw.) enthalten; hierin liegt der Grund für die Popularität von Java begründet. Der Browser lädt den Bytecode eines speziellen Java-

Programms, einem Applet, über das Internet auf den lokalen Rechner und führt ihn dort aus. Da der auf der jeweiligen Hardware lauffähige Maschinencode erst zur Laufzeit vom Interpreter erzeugt wird, ist die Ausführungsgeschwindigkeit von derartigen Javaprogrammen erheblich niedriger als die bereits kompilierter Programme. Falls auf die Ausführungsgeschwindigkeit besonderen Wert gelegt wird, so sind seit kurzem sog Just-In-Time-(JIT-)Compiler für Java verfügbar, die den Bytecode zur Ladezeit in Maschinencode übersetzen.

Java ist objektorientiert

Java beruht auf einem streng objektorientierten Konzept. Mit der (sehr rudimentären) Java-Entwicklungsumgebung JDK (Java Development Kit), wie sie momentan für sehr viele Betriebssysteme und damit Hardwareplattformen zur Verfügung steht (Windows 95/NT, IBM OS/2 und AIX, Sun Solaris, Linux usw.), werden die sog. Packages geliefert. Diese Packages sind die zu Java gehörigen Klassenbibliotheken, selbst in Java geschrieben und damit ebenso portabel wie die Programmiersprache, die zur Softwareentwicklung in Java notwendig sind. Die enthaltenen Klassen realisieren auf einem abstrakten Niveau Benutzeroberflächen, Netzwerkfunktionen, Internet-Protokolle usw. Ihre Funktionalität kann durch die Ableitung eigener von diesen vorgefertigten Klassen erweitert werden.

Sicherheit mit Java

Wird der Bytecode eines Java-Programms mit Hilfe eines javafähigen WWW-Browsers über das Internet auf den eigenen Rechner geladen und ausgeführt, so muß sichergestellt sein, daß dieses Programm auf dem lokalen Rechner keinen Schaden anrichten kann. Dies ist eine sehr wichtige Forderung, da der Benutzer zunächst keine Kontrolle darüber hat, was das Programm auf seinem Rechner wirklich macht.

Ein Sicherheitsgesichtspunkt in Java ist der Verzicht auf Zeiger und damit auch auf die in C bzw. C++ verbreitete Zeigerarithmetik, die oft zu schwer kontrollierbaren Fehlern führt. (Der Verzicht auf Zeiger hat allerdings seinen eigentlichen Grund in der Plattformunabhängigkeit; der Sicherheitsaspekt ist eher ein angenehmer Nebeneffekt.)

Entscheidend ist die Unfähigkeit von Applets (über das Internet geladene Java-Programme), auf die lokalen Daten zuzugreifen; ein Applet kann nur auf Daten zugreifen, d.h. Daten lesen und schreiben, die sich auf dem Rechner befinden, auf dem auch das Applet selbst liegt.

7.1.2. Das Zielsystem

Mit der Verwendung von Java als Programmiersprache erübrigt sich die Wahl einer Hard- und Softwareplattform.

Die Implementierung des Entwurfs erfolgte auf einem PC mit OS/2 Warp Version 4 sowie auf einer Sun Workstation unter Solaris 2.4.

7.1.3. Das relationale Datenbanksystem DB2

DB2 ist ein Verwaltungssystem für relationale Datenbanken, das dem Benutzer das Erstellen, Aktualisieren und Steuern relationaler Datenbanken mit Hilfe von SQL (*Structured Query Language*) ermöglicht. Die Konzeption von DB2 ermöglicht dabei die Erfüllung der Informationsanforderungen kleiner wie großer Anwendungen. Das Produkt ist auf einer Vielzahl von Plattformen - Großsysteme wie MVS/ESA, VM, VSE, mittlere Systeme wie OS/400, AIX sowie anderen auf Unix basierende Systeme und auf Einzelplatz- oder auf LAN basierenden Systeme wie OS/2, DOS und Windows - verfügbar.

DB2 ist ein "offenes" System: Zusätzlich zu den von IBM, dem Hersteller dieses Produkts, bereitgestellten Client-Plattformen sind DB2-Datenbankserver für den Zugriff durch andere Produkte offen, die das DRDA-Protokoll (Architektur einer verteilten relationalen Datenbank) unterstützen.

Ab der zur Zeit aktuellen Version 2.1.2 unterstützt DB2 neben der ODBC- auch die JDBC-Schnittstelle, die im nächsten Abschnitt näher besprochen werden soll.

7.1.4. Zugriff von Java-Programmen auf relationale Datenbanken

Die Firma JavaSoft, ein Tochterunternehmen von Sun Microsystems, der die Entwicklung von Java und der Java-nahen Software übertragen wurde, hat mit JDBC (*Java Database Connectivity*) eine Spezifikation für eine produktunabhängige Datenbankschnittstelle für Java zum Zugriff auf entfernte Datenbanken definiert, die zur Zeit in der Version 1.10c verfügbar ist. JDBC unterstützt die grundlegende SQL-Funktionalität; diese wird vorausgesetzt und daher hier nicht näher ausgeführt.

JDBC basiert auf dem X/Open SQL CLI (*Call Level Interface*) und lehnt sich an ODBC (*Open Database Connectivity*, Microsoft) an, dem Standard für Datenbankzugriffe im PC-Bereich. ODBC selbst kann nicht als Schnittstelle dienen, da es eine C-Schnittstelle ist, d.h. ODBC erfordert Funktionalitäten, die in Java nicht (leicht) möglich sind; daher wurde eine eigene Schnittstelle für Java entwickelt. Die Anlehnung an ODBC vereinfacht aber den Umstieg.

7.2. Realisierung des Entwurfs

Im den folgenden Abschnitten soll die Implementierung der einzelnen Komponenten des Entwurfs, also des Verwaltungsprogramms, der Anwendungs- und der Datenbankschnittstelle beschrieben werden.

7.2.1. Das Verwaltungsprogramm

Für das Verwaltungsprogramm der Organisationsdatenbank ist die Realisierung als Applet sinnvoll, da der Aufenthaltsort des Benutzers mit der Verfügbarkeit eines javafähigen Browsers nur leicht eingeschränkt ist. Als interaktives Programm benötigt das Verwaltungsprogramm eine Benutzerschnittstelle.

Entwicklung von GUIs in Java

Zur Entwicklung von graphischen Benutzerschnittstellen (*Graphical User Interface, GUI*) wird das Java-Package AWT (*Abstract Window Toolkit*) verwendet. Die Klassen des AWT kapseln das native (systemeigene) GUI einer Plattform vom Quellcode eines Javaprogramms ab:

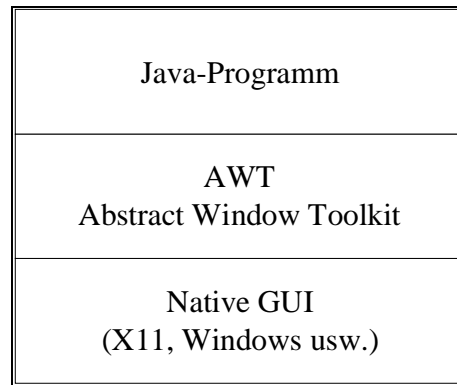


Abbildung 7.1. Funktion von AWT und GUI

Die Klassen des AWT übernehmen gewissermaßen die Rolle eines Mittlers zwischen der Anwendung und dem systemspezifischen GUI.

Prinzipiell benötigt man einen sog. *Container*, der seinerseits *Komponenten* (Elemente einer Benutzeroberfläche) enthalten kann, um eine Benutzeroberfläche wie z.B. ein Auswahlfenster zu realisieren. Die Klasse *Component*, von der auch die Klasse *Container* abgeleitet ist, stellt eine Grundfunktionalität zur Verfügung, wie z.B. Eventhandling usw. Folgende GUI-Komponenten sind derzeit (Version 1.0.2) im Lieferumfang des JDK enthalten:

- Button (Schaltfläche)
- Choice (Auswahl)
- Checkbox (Kontrollkästchen)
- CheckboxGroup (Kontrollkästchengruppe, Radiobuttons)
- List (Liste)
- Label (Textfeld)
- TextField (Texteingabefeld)
- TextArea (Texteingabegebiet)
- Canvas (Zeichenleinwand)
- Menu (Menü)
- Frame (Fenster)

Damit sind alle wesentlichen Elemente eines GUIs vorhanden.

Die Plattformunabhängigkeit der Programmiersprache und damit auch der in ihr geschriebenen Programme wirft die Frage nach der Darstellung graphischer Elemente unter den verschiedenen Systemen auf. Ein Fenster sieht unter X-Windows anders aus als unter OS/2; auch die Positionierung auf dem Bildschirm ist schwer zu beeinflussen, da die GUI-

Elemente unter den verschiedenen Systemen z.B. unterschiedliche Größen haben können. Daher führt Java sog. *Layout Manager* ein, die die GUI-Elemente abhängig von der Fenstergröße, Form und einigen Variablen selbständig plazieren.

Das GUI des Verwaltungsprogramms

Für das Verwaltungsprogramm werden verschiedene Eingabe- und Auswahlmasken benötigt. Die Bedienung des Programms geschieht in drei Stufen:

- Auswahl der zu bearbeiteten Tabelle (unterteilt in Stammdaten und Beziehungen)
- Auswahl der beabsichtigten Aktion (Hinzufügen, Ändern oder Löschen)
- Durchführen der gewählten Aktion

Das Startfenster des Programms enthält die entsprechenden Menüpunkte zur Auswahl der zu bearbeiteten Tabelle. Darüber hinaus enthält es einen Menüpunkt "Datenbank", mit dem eine Verbindung zur Organisationsdatenbank hergestellt werden kann. Da diese als Klasse mit einer statischen Variable für die Verbindung realisiert wird, kann auf diese Weise kontrolliert eine Verbindung hergestellt werden, so daß diese nicht für jede einzelne Anfrage aufgebaut zu werden braucht.

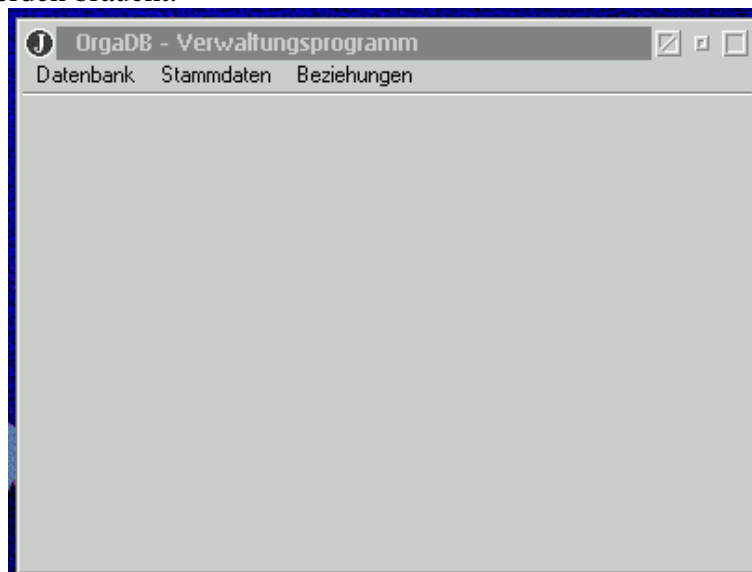


Abbildung 7.2: Startfenster des Verwaltungsprogramms

Wird eine Tabelle ausgewählt (unter "Stammdaten" sind die Entity-, unter "Beziehungen" die Relationstabelle zu finden), so wird das Aktionsauswahlfenster geöffnet, das aus einem Radiobutton besteht (Abbildung 7.3). Nachdem die gewünschte Aktion ausgewählt ist, erscheint nach Betätigung des "OK"-Buttons die entsprechende Eingabe- bzw. Auswahlmaske.

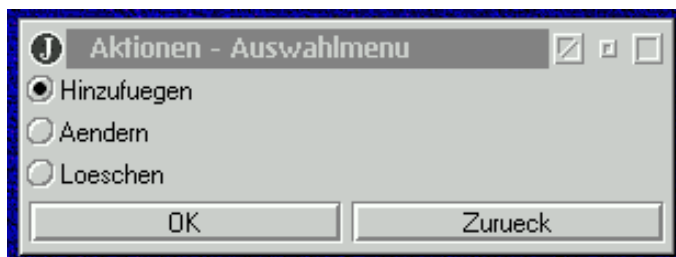


Abbildung 7.3: Aktionsauswahlfenster

Sind die gewünschten Eintragungen in der Maske vorgenommen, so werden die Daten durch Anklicken des "Ausführen"-Buttons in die Datenbank übertragen. Das Fenster wird durch die Betätigung der "Zurück"-Schaltfläche geschlossen.

Implementierung

Die Frame-Klasse des Java-AWT-Package realisiert Standardprogrammfenster, d.h. diese Fenster enthalten automatisch eine Titelseite und eine Menüzeile. Sie können um die oben beschriebenen GUI-Elemente ergänzt werden.

Auf diese Weise wird die Benutzeroberfläche des Verwaltungsprogramms realisiert; eine Klassen hierarchie wird in Abbildung 7.4 dargestellt.

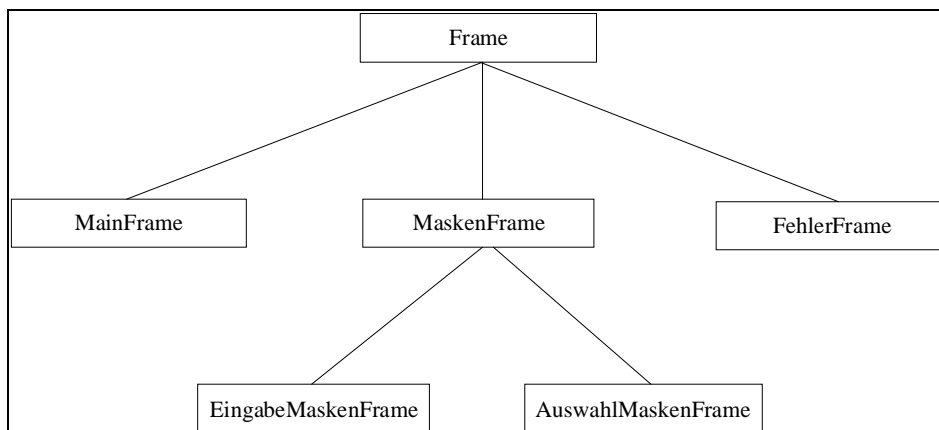


Abbildung 7.4: Klassenhierarchie der Fenster der Benutzerschnittstelle

Durch die Auswahl eines Datensatzes werden Objekte der an der Operation beteiligten Entitäten erzeugt; durch das Anklicken des "Ausführen"-Buttons werden die Methoden der Objekte aufgerufen, die ihrerseits die entsprechenden Funktionen der Organisationsdatenbankschnittstelle zur Ausführung der gewählten Operation mit den korrekten Daten aufrufen. Auf diese Weise wird die mehrstufige Form der Kommunikation des Verwaltungsprogramms mit der Datenbank implementiert (Abbildung 4.1).

Beispiel

Anhand des folgenden Beispiels soll der interne Ablauf der Organisationskomponente verdeutlicht werden:

Es ist ein neuer Mitarbeiter eingestellt worden, dessen Daten nun in die Organisationsdatenbank aufgenommen werden sollen. Der Benutzer des Verwaltungsprogramms stellt zunächst die Verbindung zur Datenbank her (Menüpunkt Datenbank | Verbindung aufbauen). Dann wählt er unter dem Menü Stammdaten den Menüpunkt Person. Auf dem erscheinenden Aktionsauswahlfenster markiert er "Hinzufügen", woraufhin die Eingabemaske für Personen geöffnet wird.

Intern werden ein "leeres" Objekt der Klasse Person und ein Objekt der Klasse Organisationsdatenbank erzeugt. Letzteres "kennt" die Parameter der bestehenden Verbindung zur Datenbank, da diese als Klassenvariablen deklariert sind, so daß alle Objekte dieser Klasse sich diese Variable (physisch) teilen.

Har der Benutzer die Daten des neuen Mitarbeiters eingetragen, so betätigt er den "Ausführen"-Button. Zunächst wird geprüft, ob alle benötigten Daten eingetragen wurden (Nachname und Geburtsdatum), Dann wird das noch leere Objekt mit den Personendaten gefüllt und die Methode `anlegen` aufgerufen, die ihrerseits die Methode `insertPerson` des Datenbankobjekts aufruft, die schließlich über einen JDBC-Aufruf das entsprechende SQL-Statement ausführt.

7.2.2. Die Anwendungsschnittstelle

Die Aufgabe der Anwendungsschnittstelle ist die Übersetzung der ihr übergebenen Suchparametern in die Funktionsaufrufe der Organisationsdatenbankschnittstelle unter Berücksichtigung der optionalen Parameter.

Dieser Vorgang wird durch Parsen des Inputstrings durchgeführt; durch die Schlüsselwörter (ATTRIBUTE usw.) kann der Charakter eines Bezugs erkannt und der dazugehörige Parameter in die entsprechende Liste übertragen werden. Die Java-Klasse *String* stellt Methoden zum Pattern-Matching zur Verfügung (*indexOf*), mit deren Hilfe eine Implementierung des Parsers effizient möglich ist. Als Datenstrukturen werden Stringfelder für die verschiedenen Kriteriumstypen benötigt.

Der Log wird als eigenes Datenbankklasse mit eigener Verbindung implementiert; diese muß bei jeder Anfrage neu auf- und abgebaut werden. Er enthält Aktivitäts-Identifikator und Bearbeiter.

7.2.4. Die Schnittstelle der Organisationsdatenbank

Die Schnittstelle realisiert die konkreten Datenbankoperationen durch die Verwendung der JDBC-API, die die Verwendung der SQL-Syntax als String in ihren Funktionen erlaubt.

Eine Datenbankoperation sieht prinzipiell folgendermaßen aus:

```
static
{
    try
    {
        // Treiber registrieren
        Class.forName("ibm.netsql.DB2Driver");
    }
    catch (ClassNotFoundException e) // Fehlerbehandlung
    {
        e.printStackTrace();
    }
}
```

```
String server = "SERVER";
String port = "PORT";
String database = "DATABASE";
String url = "jdbc:db2://" + server + ":" + port + "/" + database;
String userid = "USERID";           // Steht wirklich so im Quelltext !
String password = "PASSWORD";

try
{
    // Verbindung aufbauen
    Connection con = DriverManager.getConnection(url, userid,
password);

    // SQL-Statement initialisieren
    Statement stmt = con.createStatement();

    // Statement ausführen und Ergebnisse empfangen
    ResultSet rs = stmt.executeQuery("SELECT * FROM PERSON");

    // Ergebnisse ausdrucken
    while (rs.next())
    {
        System.out.println(rs.getString());
    }

    // Freigeben der Statement-Ressourcen
    stmt.close();

    // Schließen der Verbindung, Freigeben der Ressourcen
    con.close();
}
catch (Exception e)           // Fehlerbehandlung
{
    e.printStackTrace();
}
```

8. Verlauf der Arbeit und Ausblick

In diesem Kapitel sollen die Ergebnisse dieser Arbeit vorgestellt und über aufgetretene Probleme berichtet werden. Schließlich wird ein Ausblick auf mögliche Weiterentwicklungsmöglichkeiten gegeben.

8.1. Ergebnisse der Arbeit

Im ersten Kapitel wurde der Workflow-Management-Ansatz in den Zusammenhang mit der gesamt- und betriebswirtschaftlichen Lage der Unternehmensumgebung gestellt und damit motiviert. Eine Übersicht über die Komponenten eines WfMS ordnete die Bedeutung einer Organisationskomponente ein. Durch eine Einführung in die Aufbauorganisationsbildung wurden Erkenntnisse gewonnen, die für die Begriffsbildung von Bedeutung sind.

Im zweiten Kapitel wurden die Anforderungen, die an eine Organisationskomponente zu stellen sind, untersucht. Insbesondere stellt das WfMS als Benutzer funktionale Anforderungen an die Komponente; in diesem Zusammenhang soll sie Bearbeiter für eine Aktivität anhand von Auswahlkriterien bestimmen. Desweiteren wurde untersucht, welche Organisationsstrukturen in der Theorie der Organisationslehre und damit für die Organisationskomponente von Bedeutung sind.

Die Abtrennung der Organisationskomponente von der übrigen Ressourcenverwaltung unterstützt die verteilte Realisierung eines WfMS; die Organisationskomponente kann als einzelnes Modul realisiert und damit auch von anderen Anwendungen benutzt werden, da die Schnittstelle nur noch eine organisationspezifische Funktionalität beinhalten muß.

Im dritten Kapitel wurde ein Datenmodell entwickelt, mit dem die Abbildung der heute bekannten Organisationsstrukturen möglich ist. Es ermöglicht zudem die Erweiterung des Datenumfanges durch die Möglichkeit, neue Attribute für Personen und organisatorische Einheiten zu definieren. Der Vergleich mit einem anderen Datenmodell ergab, daß das hier vorgestellte Modell wesentlich offener bezüglich der Erweiterbarkeit ist.

In den folgenden Kapiteln wurden die Schnittstellen der Organisationskomponente, die Schnittstellen WfMS-Organisationskomponente und Organisationskomponente-Organisationsdatenbank, entworfen. Durch die Separierung der Schnittstelle zur Organisationsdatenbank konnte eine Nutzung der in ihr enthaltenen Informationen durch andere Anwendungen ermöglicht werden. Dies ist bei anderen Entwürfen nur sehr eingeschränkt möglich, da dort die Organisationskomponente meist in andere Teile des WfMS integriert ist und daher keine geeignete Schnittstelle zur Verfügung steht.

8.2. Verlauf der Arbeit

Das Gebiet des Workflow Management ist noch sehr neu und noch in der Diskussion. Daher gibt es noch sehr wenige Systeme auf dem Markt, die das ganze Spektrum der Problematik abdecken; Teillösungen sind jedoch verfügbar.

Durch den Einsatz von Java als Programmiersprache ergaben sich zeitliche Verzögerungen. Da diese Sprache ebenfalls noch sehr neu ist (die Version 1.1 der Entwicklungstools soll demnächst verfügbar sein), sind die Informationen über die Handhabung nicht immer korrekt. Weiterhin existieren für diese Sprache noch keine Entwicklungsumgebungen; dieser Umstand verzögert die Softwareerstellung erheblich. Insbesondere die Erstellung von

GUIs unter Java ist extrem zeitintensiv, da die mitgelieferten Basisklassen nur rudimentäre Funktionalität aufweisen und fertige Erweiterungen auf dem Markt noch nicht erhältlich sind. Auch diese Probleme werden aber in absehbarer Zeit gelöst sein, da die entsprechenden Produkte demnächst verfügbar sein sollen.

8.3. Ausblick

Die vorliegende Arbeit kann in Bezug auf die Auswahl eines Bearbeiters verbessert werden; der implementierte Algorithmus verwendet stets den ersten aus einer Menge von Kandidaten. Eine verbesserte Einflußnahme seitens des Benutzers wäre an dieser Stelle wünschenswert.

Als Basis für andere betriebliche Informationssysteme kann die Organisationsdatenbank weiterverwendet werden. Durch die Schaffung einer eigenen Schnittstelle sind Erweiterungen der Funktionalität auch leicht möglich.

Literatur

- [BAR96] Dr. Martin Bartonitz: Workflow Management Systeme und Business Process Reengineering. <http://www.intos.co.at/archiv>.
- [BER77] Dipl.-Hdl K Berresheim, Dipl.-Hdl Dipl.-Volksw. Dr. H. Christ, Dipl.-Hdl Dipl.-Kfm. G. Walter: Betriebliche Organisationslehre. 4. völlig neu bearbeitete Auflage. Verlag H. Starm 1977.
- [BIS84] Dipl.-Hdl. Othmar Bischoff: Organisationslehre. 17. überarbeitete Auflage. Verlag Dr. Max Gehlen 1984.
- [DOB96] Prof. Dr. Walter Doberenz: Java. Hanser 1996.
- [FRE93] Erich Frese: Grundlagen der Organisation. Konzepte - Prinzipien - Strukturen. 5. vollst. überarbeitete Auflage. Gabler 1993.
- [FRI96] David H. Friedel, Anthony Potts: Java-Programmierhandbuch. Tewi 1996.
- [GOL83] Prof. Dr. Heinz G. Golas: Organisation und Datenverarbeitung in Wirtschaft und Verwaltung. W. Giradet 1983.
- [HEI94] Prof. Dr. Heidi Heilmann: Workflow Management: Integration von Organisation und Informationsverarbeitung. In: HMD176/1994 8-21.
- [HEI96] Prof. Dr. Heidi Heilmann: Die Integration der Aufbauorganisation in Workflow-Management-Systeme. In: Prof. Dr. Heidi Heilmann, Prof. Dr. Lutz J. Heinrich, Prof. Dr. Friedrich Roithmayr (Hrsg): Information Engineering 147-165. Oldenbourg 1996.
- [HSWfMS96] Hauptseminar Workflow-Management-Systeme WS 1995/96 Fakultät Informatik, Institut für Verteilte und Parallele Höchstleistungsrechner, Universität Stuttgart
- [JAB95] Prof. Dr. Stefan Jablonski: Workflow-Management-Systeme: Motivation, Modellierung, Architektur. In: Informatik Spektrum 18(1995) 13-24.
- [JDBCa] JDBC(tm): A Java SQL API. Version 1.10. Javasoft 1996. <http://www.javasoft.com>
- [JDBCb] The JDBC(tm) API Version 1.10. Part 1 (Interfaces) und Part 2 (Classes and Exceptions). Version 1.10. Javasoft 1996. <http://www.javasoft.com>
- [JOO96] Dr. Stef Joosten: Workflow Management - Definitions. <http://www.wis.cs.utwente.nl:8080/~joosten/workflow.html#Def>.
- [KOCH96] Dipl.-Betriebswirt Olaf G. Koch, Dipl.-Betriebswirt Frank Zielke: Workflow Management. Prozeßorientiertes Arbeiten mit der Unternehmens-DV. 1. Auflage. Markt & Technik-Verlag 1996.
- [KÜH96] Ralf Kühnel: Programmierung von Java-Applets. In: Java Spektrum, Ausgaben 4-7. SIGS Conferences 1996/1997.
- [LEY96] COSA Workflow. Version 1.4.2. Produktbeschreibung. Software Ley 1996.
- [MER96] Bernhard Merkle: Der Kaffee ist fertig... In: Java Spektrum, Ausgabe 6 (November/Dezember 1996), S 29-34. SIGS Conferences 1996.
- [PIT96a] Ing. Gerald A. Pitschek: Die Organisationsdatenbank - Basis für den Workflow Management Einsatz. <http://www.intos.co.at/archiv>.

- [PIT96b] Ing. Gerald A. Pitschek: EDV-gestütztes
Geschäftsprozeßmanagement. <http://www.intos.co.at./archiv>.
- [PW94] Pfeiffer, Weiss: Lean Management. Grundlagen der Führung und
Organisation lernender Unternehmen. 2. überarbeitete und
erweiterte Auflage. E. Schmidt Verlag 1994.
- [RED96] Jens-Peter Redlich: Corba 2.0. Praktische Einführung für C++ und
Java. Addison-Wesley 1996.
- [SCH82] Günther Schanz: Organisationsgestaltung. Struktur und Verhalten.
Vahlen 1982.
- [SCHR95] Stefan Schreyjak: Anforderungsanalyse von Workflowsystemen.
Diplomarbeit Nr. 1229, Fakultät Informatik, Institut für Verteilte
und Parallele Höchstleistungsrechner, Universität Stuttgart
- [SINZ96] Prof. Dr. Elmar J. Sinz: Ansätze zur fachlichen Modellierung
betrieblicher Informationssysteme. Entwicklung, aktueller Stand
und Trends. In: Prof. Dr. Heidi Heilmann, Prof. Dr. Lutz J.
Heinrich, Prof. Dr. Friedrich Roithmayr (Hrsg.): Information
Engineering 123-143. Oldenbourg 1996.
- [VOS96] Prof. Dr. Gottfried Vossen, Prof. Dr. Jörg Becker (Hrsg.):
Geschäftsprozeßmodellierung und Workflow-Management.
Modelle, Methoden, Werkzeuge. 1. Auflage International
Thomson Publishing 1996.
- [WfMC96] The Workflow Management Coalition (Hrsg.): Terminology &
Glossary. Issue 2.0 Juni 1996. <http://www.aiai.ed.ac.uk:80/WfMC>.
- [WÖ90] Wöhe: Einführung in die ABWL. 17. überarb. und erw. Auflage.
Vahlen 1990.
- [ZEH89] Prof. Dr. C.A. Zehnder: Informationssysteme und Datenbanken.
5. durchges. Auflage. B.G. Teubner 1989.