Der rechnerunterstützte Arbeitsplatz für Führungskräfte

Habilitationsschrift

am Fachbereich 13 - Informatik
der Technischen Universität Berlin

zur Erlangung der Lehrbefähigung
für das Lehrgebiet Wirtschaftsinformatik

von
Dr.-Ing. Dipl.-Kfm.
Bodo Rieger

Habilitationsausschuß:
Vorsitzender: Prof. Dr. G. Hommel, Technische Universität Berlin
Berichter: Prof. Dr. H. Krallmann, Technische Universität Berlin
Prof. Dr. D. Friedrich, Technische Universität Berlin
Prof. Dr. P. Stahlknecht, Universität Osnabrück

Habilitationsverfahren:
Eröffnung: 24. November 1993
Fortführung: 18. Mai 1994
wissenschaftliche Aussprache: 15. Juni 1994
Aushändigung der Urkunde: 24. August 1994

Berlin, 1994

Vorwort

Die Arbeit entstand an der Nahtstelle meiner Forschungs- und Lehrtätigkeiten an den Fachbereichen Informatik der Technischen Universität Berlin und Wirtschaftswissenschaften an der Universität Osnabrück. Sie subsummiert über 10-jährige Erfahrungen und Schlußfolgerungen mit der Entwicklung rechnerunterstützter Systeme für Arbeitsplätze und Unternehmen unterschiedlichster Art. Das Spektrum reicht auf Unternehmensseite von Klein- und mittelständischen Betrieben bis hin zu multinationalen Großkonzernen, vom Fertigungs- bis zum Administrationsbereich. Die methodische Palette reicht vom einfachen Information-Retrieval über multi-mediale Informationssysteme bis hin zu modellbasierten Systemen zur Entscheidungsunterstützung, sei es klassisch optimierend, simulativ oder wissensbasiert. In nahezu allen, immer aus der Praxis abgeleiteten bzw. in der Praxis evaluierten Projekten bestätigte sich ein Bedarf zur Integration mehrerer methodischer und technologischer Ansätze, um einen angemessenen Nutzen zu erzielen.

Bei der Realisierung kamen nahezu alle Generationen von Hard- und Software zum Einsatz, vom Großrechner über PC's bis hin zu vernetzten Arbeitsplatzsystemen, von Programmiersprachen der 3. Generation über 4GL-Systeme, hierarchische und relationale Datenbanksysteme bis hin zu spezialisierter Standardsoftware zur Entwicklung von Analyse-, Planungs- und Optimierungs-Modellen sowie Experten- bzw. Führungsinformationssystemen. Schon früh setzte sich die Erkenntnis fest, daß für die Akzeptanz von rechnerunterstützten Systemen am Arbeitsplatz eine komfortable, intuitiv verständliche Benutzeroberfläche zwar notwendig ist, dies allein aber den Erfolg auf lange Sicht keineswegs hinreichend sicherstellen kann. Als entscheidend und kritisch erweisen sich vielmehr Entwicklungs- und Wartungsproduktivität. Dies gilt umso mehr, als wachsende Dynamik innerhalb und außerhalb von Unternehmen eine permanente, lebenslange Anpassung inhaltlicher und technologischer Art erforderlich machen. Hierzu zählen veränderter Informations- und Analysebedarf ebenso wie eine Offenheit für das Hinzufügen von Komponenten auf der Basis innovativer Methoden, z.B. fallbasierten Schließens, neuronaler Netze oder der Hypertext/Hypermedia-Technologie.

Entwicklungs- und Wartungsproduktivität werden entscheidend bestimmt durch Umfang und Komplexität der vorgefertigten, anwendungstyp-spezifischen Bausteine in Software-

Entwicklungs-Werkzeugen. Dementsprechend bildet der Konstrukt-Begriff das zentrale Element dieser Arbeit. Trotz erheblicher Fortschritte in diesem Bereich, die maßgeblich für den breiten Erfolg der individuellen Datenverarbeitung mit verantwortlich zeichnen, kam dabei ein Aspekt entschieden zu kurz, der gerade für Unternehmen in einer durch Arbeitsteilung und zunehmende Spezialisierung determinierten Welt kennzeichnend ist: der Aspekt der Delegation und Kooperation. Entwicklungswerkzeuge nahezu aller methodischen Schattierungen stellen jeweils ihre "eigenen" Anwendungen in den Mittelpunkt und erlauben allenfalls sporadisch, ergänzend oder hilfsweise initialisierend die Übernahme von Fremddaten; von einem Wandel des Selbstverständnisses hin zum Server-Prinzip ist nichts zu verspüren. Im Gegenteil, Datenmodell, -strukturen und -basis, der eigentliche harte, das Know-how eines Unternehmens manifestierende Kern, wird zunehmend zu einem Server-Dasein zementiert.

Neben einer fortschreitenden Entwicklung anwendungstyp-mäßig spezialisierter Bausteine (Konstrukte) in methodisch ausgerichteten und begrenzten Entwicklungswerkzeugen wird deshalb in dieser Arbeit dafür geworben, zwei Komponenten in den Mittelpunkt zu stellen: ein objektorientiertes Information-Repository, das quasi das strukturelle, funktionale und inhaltliche Wissen repräsentiert und eine Anwendungssystem-übergreifende Benutzerschnittstelle, die in der Lage ist, den Dialog mit - auch zukünftig im Rahmen methodischer Fortschritte neu auf den Markt kommenden - Analyse- und Planungssystemen zu führen. Der im letzten Teil der Arbeit vorgestellte Prototyp erhebt nicht den Anspruch einer idealen Realisierung dieses Konzepts, sondern will lediglich Argumente für dessen Praktikabilität liefern.

Analog zu den beschriebenen Potentialen einer offenen, kooperativen Systemarchitektur ist auch die vorliegende Arbeit in einem offenen, kritischen und dadurch erst kreativen Umfeld entstanden. Für ungezählte Anregungen, Experimentierfreudigkeit und konstruktive Kritik bin ich allen Mitarbeitern, Gesprächs- und Projektpartnern aus Theorie und Praxis zu Dank verpflichtet. An erster Stelle nennen möchte ich besonders die Professoren H. Krallmann und P. Stahlknecht, die aufgrund ihres unerschöpflichen Erfahrungsschatzes jederzeit für die rechte Balance zwischen Theorie und Praxis und damit dafür gesorgt haben, daß bei aller Optimierung des Weges letztendlich nicht das Ziel aus den Augen verloren gegangen ist.

Osnabrück, im November 1993

Inhaltsübersicht

Inhaltsverzeichnis

1. Einleitung

1.1. Einordnung

Die Arbeit ordnet sich als Anwendung Informatik-bezogener Erkenntnisse auf betriebswirtschaftliche Funktionen und Aufgaben allgemein in den Bereich der Wirtschaftsinformatik (WI) ein. Im besonderen befaßt sie sich mit Gesichtspunkten der Konzeption und Implementierung des Rechnerunterstützten Arbeitsplatzes (RAP) für Führungskräfte. Der RAP wird dabei in Analogie zu ([Morton 83, MSS]) aufgefaßt als eine Kombination rechnerunterstützter Komponenten, die sich aus zwei Klassen von Systemen rekrutieren:

- Die erste Klasse wird als *Decision Support Systeme* (DSS), oder Entscheidungs-Unterstützungs-Systeme (EUS) im engeren Sinne bezeichnet. Darunter werden Systeme verstanden, deren Unterstützungsfunktion primär und hauptsächlich in der Bereitstellung von (alternativen) Entscheidungsvorschlägen sowie deren Bewertung und Auswahl liegt. Sie konzentrieren sich damit eher auf die späteren Phasen von betrieblichen Entscheidungsprozessen.
- Die zweite Klasse wird als *Data Support Systeme* bezeichnet. Hierunter werden Systeme verstanden, deren Unterstützungsfunktion in der kontinuierlichen Beobachtung und Analyse von für Unternehmen relevanten, internen und externen Abläufen liegt. Sie dienen primär der Kontrolle und Überwachung mit dem Ziel, Entscheidungsprozesse im Sinne der Generierung und Ausführung konkreter Maßnahmen zu initiieren. Sie konzentrieren sich damit eher auf die frühen Phasen von betrieblichen Entscheidungsprozessen.

Das Konzept *Rechnerunterstützter Arbeitsplätze* (RAP) ist prinzipiell auf alle Typen von Arbeitsplätzen in Unternehmen ausgerichtet. Insbesondere beschränkt es sich weder auf bestimmte Ebenen in der Unternehmenshierarchie, noch auf bestimmte Zielgruppen, wie dies - historisch gesehen - meist im Zusammenhang mit konkreten Entwicklungswerkzeugen geschehen ist. Im Gegenteil setzt es sich die Integration verschiedener Hierarchieebenen im Unternehmen bezüglich Aufbau- und Ablauforganisation zum Ziel.

Nach mehreren gescheiterten Anläufen hat die Rechnerunterstützung im Führungsbereich seit Ende der 80er Jahre mit den sich etablierenden *Executive Information Systems* (EIS) eine erneute Renaissance erfahren. EIS wurden dabei ursprünglich speziell auf die Informationsbedürfnisse von (obersten) Führungskräften ausgerichtet. Da die Theorie dieser Unternehmensebene bezüglich Rechnereinsatz im Zeichen einer retrospektiven Vorgehensweise

primär einen Bedarf an Data Support-Funktionen unterstellte, wurden und werden EIS häufig synonym zu Data Support Systemen verwendet. Wie sowohl erste Einsatzerfahrungen vor allem in den USA als auch die detaillierte Analyse in dieser Arbeit zeigen, ist jedoch weder eine Beschränkung des Einsatzes dieser Werkzeuge auf die Zielgruppe der Führungskräfte sinnvoll, noch genügen ausschließlich Data Support-Funktionen einem - vor allem in die Zukunft gerichteten - Anforderungsprofil dieser Zielgruppe. Tendenziell ist vielmehr die Integration von Decision Support-Funktionen zu fordern und - im Gegensatz zu dem in dieser Arbeit vorgeschlagenen, kooperativen Konzept - in Form einer funktionalen Ausweitung bei "EIS-Werkzeuge" bereits zu beobachten.

Um in der Begriffswelt einheitlich zu bleiben, sollen im folgenden Executive Information Systems (EIS) als konkrete Ausprägung des Data Support interpretiert werden, und zwar zugeschnitten und spezialisiert auf den Data Support-*Anteil* der Arbeitsplätze von Führungskräften. Da sich die Arbeit hauptsächlich auf diese Zielgruppe konzentriert, werden daher EIS und Data Support teilweise synonym verwendet. Analog wird der allgemeinere Terminus RAP durch einen spezielleren Begriff ergänzt, der ebenfalls auf die Zielgruppe der Führungskräfte konzentriert ist: Executive Support System (ESS) (vgl. [Krallmann 87, Executive Support System]). ESS repräsentieren dann explizit die zielgruppenorientierte Kombination von Data- und Decision Support-Komponenten. Im Deutschen entspräche diese dem Terminus Führungsunterstützungs- oder Managementunterstützungssysteme (MUS). Im Rahmen dieser Arbeit wird allerdings der in der Praxis gängigere Begriff der Führungsinformationssysteme bevorzugt.

Implementierungsversuche von Führungsinformationssystemen wurden historisch mit allen Generationen von Software-Werkzeugen unternommen, angefangen bei Programmiersprachen der 3. Generation in der klassischen MIS-Ära und sogenannten Endbenutzersprachen oder 4-GL-Systemen. Neben einer zunehmenden Annäherung der Sprachkonstrukte an die Fachsprache potentieller Anwender sowie Fortschritten in Bezug auf die Benutzerfreundlichkeit durch grafische Oberflächen ist dieser Prozeß vor allem gekennzeichnet durch die Integration komplexerer, an Aufgabenstrukturen ausgerichteter Software-Konstrukte: Beispiele hierfür sind zusätzliche Datentypen wie Währung, Datum oder Zeitreihe, Analysefunktionen wie "What-if" oder "What-to-do-to-achieve" sowie mehrdimensionale Modellstrukturen mit hierarchischer Konsolidierung.

Ergebnis dieses Prozesses sind spezialisierte Software-Werkzeuge, die durch ihre Zusammenstellung dieser Konstrukte Produktivitäts- und Akzeptanzvorteile für jeweils eine Gruppe oder Klasse von Aufgaben versprechen. Bei der Konstruktzusammenstellung stehen allerdings primär methodische Gesichtspunkte im Vordergrund: Planungssprachen bzw. Expertensystemshells sind Beispiele für konventionellen bzw. wissensbasierten Decision Support, die sogenannten EIS-Generatoren stehen für den Bereich des (konventionellen) Data Support. Typische EIS-Konstrukte sind beispielsweise Exception-Reporting, Drill-Down oder Wiedervorlage.

Wenn auch in keinem Fall die jeweils anvisierte Zielgruppe der unmittelbaren Entscheidungsträger auf breiter Front überzeugt, d.h. für den direkten Einsatz am eigenen Arbeitsplatz gewonnen werden konnte, so kann doch ein nennenswert verbessertes Akzeptanzniveau in Unternehmen konstatiert werden. Zusammen mit nachweisbaren Produktivitätsgewinnen bestärkt diese Entwicklung ein Fortschreiten auf dem Weg anwendungsorientiert spezialisierter Entwicklungswerkzeuge, wie es im Bereich der Expertensysteme mit den sogenannten Applikationsshells bereits erfolgreich praktiziert wird.

Im Sinne des übergeordneten RAP-Konzeptes stellen konkrete Anwendungen, die mit diesen spezialisierten Softwarewerkzeugen realisiert werden, lediglich anteilige Unterstützungsleistungen für das Aufgabenspektrum einzelner Arbeitsplätze und auch des gesamten Unternehmens dar. Von der Qualität und Güte des Zusammenwirkens dieser Einzelkomponenten, sowohl innerhalb der Teilaufgaben einzelner Arbeitsplätze, als auch zwischen den Aufgabenspektren des Gesamtunternehmens, hängt deren eigentlicher Nutzen entscheidend ab. Die Kooperation ist dabei keineswegs ausschließlich auf die Schnittstellen zwischen rechnerunterstützten Systemkomponenten beschränkt, sondern muß an beliebigen Stellen auch rein manuelle, d.h. nicht rechnerunterstützte Teilaufgaben integrieren.

Als kritische Erfolgsfaktoren rechnerunterstützter Arbeitsplätze im Unternehmen müssen demzufolge zumindest zwei Ebenen in Betrachtung gezogen werden:

- Die *Aufgabenadäquanz* der Konstrukte der jeweils eingesetzten Softwarewerkzeuge und
- die *Integrations- und Kooperationsfähigkeit* von konkreten Anwendungen mit diesen Softwarewerkzeugen andererseits.

Eine ausführliche Analyse ausgewählter Softwarewerkzeuge aus der EIS- und der DSS-Klasse unter diesen Gesichtspunkten sowie unter Berücksichtigung der spezifischen Anforderungen der jeweiligen gesetzten Leistungsspektren offenbart trotz des hohen, erreichten Leistungsniveaus jedoch noch gravierende Defizite. Diese liegen sowohl im Bereich der verwendeten Konstrukte, die zur Repräsentation des unternehmensinternen und -externen Geschehens verwendet werden, als auch im Bereich der Vorgehensweisen beim Entwurf von RAP. Sie treten insbesondere mittel- bis langfristig gesehen, also mit zunehmender Lebensdauer von RAP-Systemen in Erscheinung.

Angesichts immer komfortablerer Entwicklerschnittstellen von Softwarewerkzeugen spielen nicht mehr die Erst-Entwicklungszeiten und -kosten die entscheidende Rolle, sondern Aspekte der Wartung, Anpassung und Weiterentwickelbarkeit treten in den Vordergrund. Dabei wird das Problem aus zwei Richtungen verschärft: zum einen weitet sich der Kreis von Systementwicklern durch zunehmende Verlagerung in die Fachabteilungen permanent aus, zum anderen nimmt die Entwicklungsproduktivität, gemessen in der Anzahl generierter Anwendungssysteme pro Zeiteinheit zu. Herkömmliche Strukturen und Mechanismen zur notwendigen Steuerung, Standardisierung und Überwachung im Sinne der Aufrechterhaltung der Kommunikationsfähigkeit der realisierten Systeme verlieren zunehmend an Wirksamkeit. Der kurzfristige Nutzen schnellerer und besserer weil lokaler, rechnerunterstützter Problemlösung steht in Gefahr mittel- und langfristig ins Gegenteil umzuschlagen in Form einer Flut inkompatibler, redundanter, unüberschaubarer, nicht mehr wartbarer lokaler Anwendungssysteme.

1.2. Schwerpunkte

Ziel dieser Arbeit ist es daher, neue Software-Konstrukte und Konstrukt-Kombinationen zu identifizieren bzw. zu entwerfen, die einen Beitrag zu einer höheren Wartungsfreundlichkeit rechnerunterstützter Arbeitsplätze leisten können. Die neuen Konstrukte werden dabei aus den innovativen Informatik-Bereichen der objektorientierten Programmierung, des computergestützten Software-Engineering, speziell der Entwicklungsdatenbanken (Repositories), des kooperativen Problemlösens, der multi-medialen Informationsverarbeitung sowie der künstlichen Intelligenz abgeleitet. Die Anwendung der Konstrukte bezieht sich sowohl auf Ergänzungen in einzelnen Werkzeugklassen als auch auf die Herstellung der Kooperationsfähigkeit zwischen Anwendungsteilen, die mit diesen Werkzeugklassen implementiert werden.

Als Konsequenzen ergeben sich nicht nur konzeptionelle Neuroientierungen bei der Architektur von Entwicklungswerkzeugen, sondern auch veränderte Anforderungsprofile im Bereich der Unternehmensorganisation sowie der Personalqualifikation. Auf die Herleitung und Begründung dieser Konsequenzen zielen die Anforderungsanalyse in Teil 1, die Schwachstellenanalyse in Teil 2 und das Integrationskonzept in Teil 3 dieser Arbeit.

1.3. Aufbau der Arbeit

Die Arbeit gliedert sich in drei Teile mit je drei Kapiteln. Teil I befaßt sich mit den Anforderungen an Funktionalität und Struktur rechnerunterstützter Arbeitsplätze für Führungskräfte. In Kapitel 2 werden zunächst Unterstützungspotentiale auf der Basis von Untersuchungen

über typische Aufgaben und deren Anteile im Führungsbereich herausgearbeitet. Es dient der Abgrenzung des überhaupt einer Rechnerunterstützung zugänglichen Aufgabenspektrums. Dabei werden sowohl potentielle Reorganisationen des Arbeitsablaufs aufgrund technologischer Innovationen einbezogen, als auch Seiteneffekte organisatorischer Art berücksichtigt. Letzteres führt insbesondere zu der Erkenntnis, daß eine Beschränkung der Betrachtung auf eine, die oberste Unternehmensebene nicht ausreichend ist. Vielmehr müssen alle Führungs- und Entscheidungsebenen im Unternehmen im Hinblick auf eine Straffung der Unternehmenshierarchie (Lean Management) einbezogen werden. Kapitel 3 stellt diesem Anforderungsprofil eine Analyse des Leistungsspektrums der speziell für den Anwenderkreis der Führungskräfte konzipierten Executive Information Systems gegenüber. Aus den Diskrepanzen dieser Gegenüberstellung, insbesondere in Bezug auf die organisatorische Begrenzung auf die oberste Unternehmensebene sowie die inhaltliche Konzentration auf Funktionen des Data Support, wird in Kapitel 4 ein Strukturkonzept für rechnerunterstützte Arbeitsplätze abgeleitet und soweit verallgemeinert, daß es geeignet ist, prinzipiell alle Unternehmensebenen und Aufgabenklassen homogen zu repräsentieren. Es basiert auf einer Kombination von Anwendungssystemklassen, die nach dem Kriterium der Entscheidungsnähe in Entscheidungsprozesse initiierende (EIS) und konkrete Entscheidungsalternativen vorbereitende Systeme (DSS) unterscheidet.

Teil II der Arbeit befaßt sich mit der Darstellung und Analyse der Konstrukte spezialisierter Softwarewerkzeuge für diese beiden Systemkategorien. Unter Konstrukten werden dabei vorgefertigte Softwarebausteine verstanden, aus denen durch Kombination und Parametrisierung konkrete Anwendungen zusammengestellt werden können. Für die Analyse werden jeweils Referenzmodelle entwickelt, die die wesentlichen strukturellen und funktionalen Elemente herausstellen. Kapitel 5 konzentriert sich auf den Bereich der Entwicklungswerkzeuge für Führungsinformationssysteme, im folgenden als EIS-Generatoren bezeichnet, und unterscheidet Konstruktgruppen zur Verwaltung, Selektion, Präsentation und Verarbeitung von Information. Kapitel 6 führt eine methodisch orientierte Trennung der Entwicklungswerkzeuge für Entscheidungs-Unterstützungs-Systeme in konventionelle und wissensbasierte DSS-Generatoren durch. Gleichwohl wird für beide das gleiche Referenzmodell mit den Kategorien Modell- und Anwendungs-Spezifikation sowie Modellberechnung angewandt. Das zusammenfassende Kapitel 7 schließlich arbeitet auf dieser Basis die systemimmanenten Leistungsgrenzen einer Realisierung des verteilten RAP-Konzepts des ersten Teils heraus und stellt damit gewissermaßen das Anforderungsprofil für den folgenden dritten Teil dar.

Teil III befaßt sich mit innovativen Ansätzen der Informatik zur Repräsentation von Problemstrukturen und -lösungsprozessen sowie deren möglichem Beitrag zur Lösung der im zweiten Teil abgeleiteten Engpässe. Kapitel 8 präsentiert und analysiert relevante Konstruk-

te, die sich an der Schwelle zur Umsetzung in konkrete Entwicklungswerkzeuge und damit in die Praxis befinden. Es sind dies im einzelnen die objektorientierte Programmierung, das Information Repository, verteilte Systeme, Hypertext/Hypermedia und neuronale Netze. Kapitel 9 entwickelt unter Berücksichtigung der Konstrukte dieser Ansätze ein Struktur- und Vorgehensmodell für eine integrative, verteilte Architektur für rechnerunterstütze Arbeitsplätze und demonstriert dessen Funktions- und Wirkungsweise an einem Protyp-System (proFOUND-Manager). Kapitel 10 schließlich leitet Konsequenzen für Aufbau- und Ablauforganisation sowie Personalqualifikation ab und formuliert Perspektiven für die Entwicklung von Standardsoftware. Diese können bereits heute als Beurteilungs- und Auswahlkriterien für Entwicklungswerkzeuge für Führungsinformationssysteme interpretiert werden.

Teil I

Der Arbeitsplatz von **Führungskräften**

2. Der Arbeitsplatz von Führungskräften

2.1. Klassifikationsmodelle für Aufgaben, Tätigkeiten und Werkzeuge

2.1.1. Forschungsansätze

Die Frage, wo und wie die Arbeit von Führungskräften durch rechnergestützte Werkzeuge erleichtert bzw. verbessert werden kann, setzt Wissen über deren Aufgaben, Tätigkeiten sowie derzeit eingesetzte Hilfsmittel voraus. Bei der wissenschaftlichen Ermittlung und Beschreibung dieses Wissens können im groben zwei Richtungen beobachtet werden:

- Die historisch ältere *normative, präskriptive* Managementlehre und
- die modernen, quantiativen, mehr verhaltens- und sozialwissenschaftlich orientierten Ansätze der *empririschen* Managementforschung.

Aus beiden Richtungen entstanden klassifikatorische Modelle über die Aufgaben und Tätigkeiten von Führungskräften, die im folgenden kurz dargestellt und bezüglich ihrer Eignung für die Ableitung von Anforderungen an die Ausstattung und Gestaltung rechnergestützter Werkzeuge sowie deren Integration in den Managementalltag bewertet werden.

2.1.2. Ansätze der normativen Managementlehre

Wenngleich die Ursprünge weiter zurückreichen, werden die von Mintzberg als "Classical School" bezeichneten Arbeiten von Fayol allgemein als erste nennenswerte Publikationen der Ausprägung normativer, präskriptiver Managementlehre angesehen. Sie kulminieren in der Feststellung von fünf Funktionsbereichen, die jedoch eher als allgemeine Prinzipien gedeutet werden (vgl. [Fayol 49, Management]; [Mintzberg 73, Managerial Work], S. 8)):

- (Voraus-)Planen
- Organisieren, speziell betrieblicher Ressourcen
- Koordinieren
- Anweisen und
- Kontrollieren.

Eine auf Fayol's funktionaler Analyse aufbauende, verfeinerte Gliederung stammt von Gulick, der hierfür das Kürzel POSDCoRB einführte (vgl. [Gulick 37, Theory of Organization]):

- *Planning* als zielgerichtetes, grobes Planen, welche Aktivitäten mit welchen Mitteln und Methoden Gegenstand des Unternehmens sein sollen, z.B. Herstellung und Verkauf integrierter Softwarepakete unter MS-Windows,
- *Organizing* als Organisieren der Aufbau- und Ablaufstruktur des Unternehmens, z.B. Spartengliederung nach dem Profit-Center-Prinzip,
- *Staffing* als Personalpolitik bezüglich Beschaffung, Qualifikation und Arbeitsbedingungen, z.B. Fertigung in autonomen Arbeitsgruppen,
- *Directing* als das Führen der Mitarbeiter durch das Treffen und Umsetzen von Entscheidungen in konkrete Handlungsanweisungen bzw. Zielvorgaben, z.B. Entwickeln einer neuen Produktkomponente bis zu einem vorgegebenen Termin,
- *Coordinating* als übergeordnetes Abstimmen der Arbeitsgruppen im Sinne von zeitlichen Meilensteinen und inhaltlichen Kompetenzen, z.B. rollierende, monatliche Fertigungsplanung für das Folgequartal auf der Basis entsprechender Absatzpläne,
- *Reporting* als regelmäßige, überwachende Berichterstattung an über- und untergeordnete Stellen incl. seiner selbst, z.B. über Plan-Ist-Status oder interne und externe Entwicklungen, sowie
- *Budgeting* als Erstellung operativer Planvorgaben, z.B. Mittelzuweisungen, Absatz-, Umsatz- und Kostenzielen, aber auch Wachstumsraten oder Produktivitäten und Kapazitätsauslastungen.

Den genannten und zahlreichen, darauf aufbauenden, bis in die Gegenwart reichenden Arbeiten (vgl. [Dale 72, Management]; [Drucker 67, Ideale Führungskraft]; [Drucker 77, Management]) wird von Seiten der empirischen Managementforschung "angelastet", daß sie idealtypisch, theoretisierend und zum Teil mythologisierend überhöht sind. Insbesondere können sie aufgrund des verwendeten Abstraktionsniveaus keine konkreten Ansatzpunkte für der Rechnerunterstützung zugängliche (Einzel-)Aktivitäten liefern.

Weitere von Mintzberg beschriebene Ansätze der Managementforschung können ebenfalls kaum brauchbare Beiträge in dieser Richtung liefern, da sie sich jeweils auf ausgewählte Aufgabenkomplexe bzw. Rollen von Führungskräften, speziell die Entscheidungsfindung und die Führungsrolle, konzentrieren (vgl. [Mintzberg 73, Managerial Work], S. 12-21):

- Zu den *entscheidungs-orientierten* Ansätzen zählt er die mikroökonomische Theorie des rational handelnden, ausschließlich Gewinn maximierenden Unternehmers (Entrepreneur School), sowie die Versuche auf der Basis der Arbeiten von Simon, schlecht-strukturierte Entscheidungsfindung zu modellieren und zu automatisieren (Decision Theory School). Auf letztere geht immerhin das Konzept der Entscheidungs-Unterstützungs-Systeme (Decision Support Systems) zurück (vgl. Kapitel 3).
- Die Ansätze mit Betonung der *Führungsrolle* unterteilt Mintzberg in Arbeiten über Führungsstile (Leader Effectiveness School), Potentiale und Einsatzformen von Macht (Leader Power School) sowie "Verhaltens-"Studien (Leader Behaviour School). Letztere beschränken sich jedoch auf die Auswertung subjektiver Sichtweisen und Beschreibungen der Führungskräfte selbst und gehen in den gewonnenen Klassifizierungen wenig über das Niveau der klassischen Schule (POSDCoRB) hinaus.

2.1.3. Ansätze der empirischen Managementforschung

Im Gegensatz zu den normativen Ansätzen, die sich an den Fragen "Was ist Management?" bzw. "Was soll Management sein?" orientieren, stellt die induktive, empirische Managementforschung die Frage "Was tut Management?" in den Vordergrund. Mit dem Wechsel der Betrachtung, weg von einer allgemeinen Führungsaufgabe, hin zur Ableitung konkreter Führungstätigkeiten aus der Praxis, entsteht die Notwendigkeit einer genaueren Abgrenzung der zu untersuchenden Zielgruppe "Führungskraft". In Anlehnung an Vallone (vgl. [Vallone 91, computergestützte Arbeitsumgebung], S. 18) und Hales (vgl. [Hales 86, What do managers do?], S. 89) wird im folgenden die Definition von Stewart verwendet, wonach als *"Führungskräfte alle Mitarbeiter inner- oder außerhalb von Linie und Stab oberhalb einer Ebene etwa des Vorarbeiters"* angesehen werden (vgl. [Stewart 76, Contrasts in Management], S. 4).

Beispiele für Studien dieser von Mintzberg als "Work Activity School" bezeichneten Richtung reichen bis in die 50er-Jahre zurück (vgl. [Carlson 51, Executive Behaviour]; [Stewart 67, Managers and their Jobs]). Mintzberg listet allein 14 Studien bis 1967 auf (vgl. [Mintzberg 73, Managerial Work], S. 23). Bezüglich der Ergebnisse muß zwischen inhaltlichen und attributiven Aspekten unterschieden werden.

Attributive Aspekte betreffen Art, Anzahl, Dauer und eingesetzte Hilfsmittel der Aktivitäten, z.B. 5-minütiges Telefonat. Diese Ergebnisse bieten gute Ansatzpunkte für Überlegungen substitutiver Verbesserungen durch (innovative) rechnergestützte Werkzeuge.

Während diese (attributiven) Fakten relativ leicht zu erheben und zu kategorisieren sind, treten bei den inhaltlichen Aspekten ähnliche Probleme wie in den klassischen Ansätzen auf. Viele der (frühen) Studien gehen deshalb auch kaum über die POSDCoRB-Klassifizierung hinaus. Mintzberg verfolgte in seiner "klassischen" Studie den Ansatz, die in der Erhebungsphase mittels Beobachtung "frei" nach dem augenscheinlichen Zweck benannten Aktivitäten erst nachträglich zu kategorisieren. Als Hauptergebnis gelangt er zu einer Theorie von 10 Rollen, die er in folgende 3 Kategorien zusammenfaßt:

- *Interpersonelle Rollen*: Repräsentieren, Führen und Liieren.
- *Informationelle Rollen*: Beobachten, Verteilen und Sprechen.
- *Entscheidungsrollen*: Unternehmen, Störungen regeln, Ressourcen zuordnen und Verhandeln.

Obwohl Mintzberg damit im wesentlichen die Ungültigkeit der klassischen Kategorisierung empirisch belegt, liefert seine Klassifizierung dennoch wenig Hinweise für Unterstützungspotentiale für den Rechnereinsatz. Noch weniger gilt dies für die Arbeiten von Kotter, die vor allem die Bedeutung der politischen Dimension herauskristallisieren und in den beiden Klassen "Agenda setting" und "Network building" münden (vgl. [Kotter 82, The general managers]).

Mehr in Richtung Aktivitäten-orientiertes Referenzmodell zielt das sogenannte "Leader Observation System (LOS)-Modell", das aus einer Studie von Luthans/Lookwood hervorgegangen ist (vgl. [Luthans 84, Measuring leader behaviour]). Danach wird vereinfachend folgendes Aufgabenprofil gesehen:

- Planung und Koordination
- Personalbeschaffung
- Aus- und Weiterbildung
- Entscheidung und Problemlösung
- Schreibarbeit
- Austausch von Routineinformationen
- Überwachung und Kontrolle der Leistung
- Motivation und Verstärkung
- Disziplinarische Maßnahmen und Bestrafung
- Interaktion mit anderen

- Konfliktbewältigung
- Gesellschaftliche und politische Aktivitäten.

Detailliertere und mehr für die Analyse von Unterstützungspotentialen geeignete Klassifizierungen finden sich in einzelnen Studien. Beispielhaft wird nachfolgend die Gliederung von Engel et al. angeführt, da sie die Grundlage einer der in Kapitel 2.2 ausführlich zitierten Studien bildet (vgl. [Engel 79, Office communications]). Ferner qualifiziert sich diese Studie in Anlehnung an Vallone dadurch, daß sie sich neben 3 Gruppen von Führungskräften zu Vergleichszwecken auch auf Sekretariatspersonal sowie Unterstützungskräfte erstreckt. Die Gruppierung der Einzelaktivitäten ist von Vallone übernommen (vgl. [Vallone 91, computergestützte Arbeitsumgebung], S. 27 ff.):

- *Kommunikation*
  Telefonieren, geplante und ungeplante Besprechungen, Post, Konferieren mit Sekretariat bzw. Vorgesetzten, Reisen außer Haus
- *Schriftguthandhabung*
  Schreiben, Lesen, persönliches oder maschinelles Diktat, Öffnen und Versenden
- *Informationsverarbeitung*
  Archivieren, Informationsabruf, Kopieren, Rechnen, Korrekturlesen, Suchen, Terminkalender führen, Sammeln und sortieren
- *Sonstiges*
  (Termin-)Planung, Equipment nutzen, sonstiges.

2.2. Ergebnisse empirischer Studien

2.2.1. Zeitverteilung von Tätigkeiten

Die Untersuchung der Zeitanteile verschiedener Aktivitäten in Verbindung mit den dazu eingesetzten (technischen) Hilfsmitteln liefert wesentliche Hinweise für das Nutzenpotential substitutiver innovativer Werkzeuge. Auffälliges Ergebnis nahezu aller Zeitstudien ist der übereinstimmend hohe Anteil kommunikativer Tätigkeiten und darunter die Dominanz direkter Kommunikation "von-Angesicht-zu-Angesicht" bzw. per Telefon. Trotz unterschiedlichster Erhebungsmethoden (Tagebuchmethode, Beobachtung, Interview) decken sich die Aussagen der meisten Studien in diesem Bereich. Die Tabelle in Abb. 2-1, basierend auf einer Studienübersicht von Panko (vgl. [Panko 88, Enduser Computing], S. 14), sowie die daraus von Vallone abgeleitete, summarische Darstellung in Abb. 2-2 (vgl. [Vallone 91, computergestützte Arbeitsumgebung], S. 25) dokumentieren diese Feststellung.

Studie	Anteil Kommunikationstyp direkt	indirekt	sonst.
Teger	56	29	15
Stogdill & Shartle	66	k.A.	k.A.
Engel et al.	40	34	26
Stewart	60	28	12
Hinrichs	53	25	22
Burns	52	24	24
Horne & Lupton	63	24	13
Doktor	78	k.A.	k.A.
Dubin & Spray	61	5	34
Croston & Goulding	63	18	19
Ives & Olson	78	k.A.	k.A.
Mintzberg	70	k.A.	k.A.
Kurke & Aldrich	70	k.A.	k.A.
Kelley	61	k.A.	k.A.
Choran	53	k.A.	k.A.
Palmer & Belshon	60	15	25
Notting	59	17	24
Durchschnitt	ca. 59	ca. 21	ca. 20

Abb. 2-1: Kommunikationsaktivitäten im Managementbereich: Studienübersicht (Quelle: [Panko 88, Enduser Computing], S. 14)

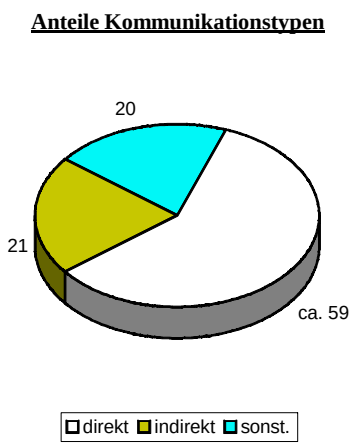


Abb. 2-2: Anteile direkter und indirekter Kommunikation im Management (Quelle: [Vallone 91, Computergestützte Arbeitsplatzumgebung für das Management], S. 25)

Aussagen über die Zeitanteile der Kommunikation sowie anderer Tätigkeitsgruppen incl. der zugehörigen Einzeltätigkeiten lassen sich der Studie von Engel et al. entnehmen (vgl. [Engel

79, Office communications], S. 403-406). Abb. 2-3 zeigt diese Ergebnisse im Vergleich der

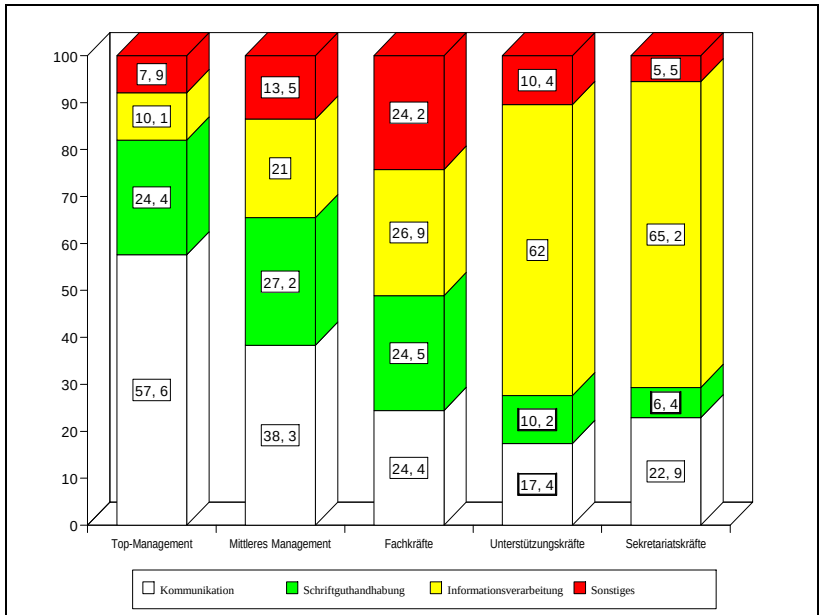


Abb. 2-3: Zeitverteilung der Tätigkeit(sgrupp)en je Mitarbeitergruppe (Quelle: [Engel 79, Office Communications], S. 403 ff.)

Mitarbeitergruppen.

Die Studie basiert auf einer Fragebogenaktion von über 1700 Mitarbeitern eines dezentral organisierten, multinationalen Unternehmens der Konsumgüterbranche. Es kann (erwartungsgemäß) festgestellt bzw. bestätigt werden, daß Kommunikation mit zunehmender Führungsfunktion dominiert. Diese Aussage wird noch verstärkt durch eine Untersuchung von Klemmer/Snyder bezüglich des Einflusses der angewandten Untersuchungsmethode. Sie stellten fest, daß bei Fragebogenerhebungen im Vergleich zur Beobachtungsmethode der Zeitaufwand für (direkte) Kommunikation tendenziell unterschätzt, derjenige für lesende und schreibende Schriftgutbearbeitung sowie Informationsverarbeitung dagegen überschätzt wird (vgl. [Klemmer 72, Time spent communicating]).

Die folgenden Abbildungen Abb. 2-4 bis 2-6 geben einen tieferen Einblick in die Verteilung der einzelnen Tätigkeiten der Tätigkeitsgruppen auf die Beschäftigtengruppen.

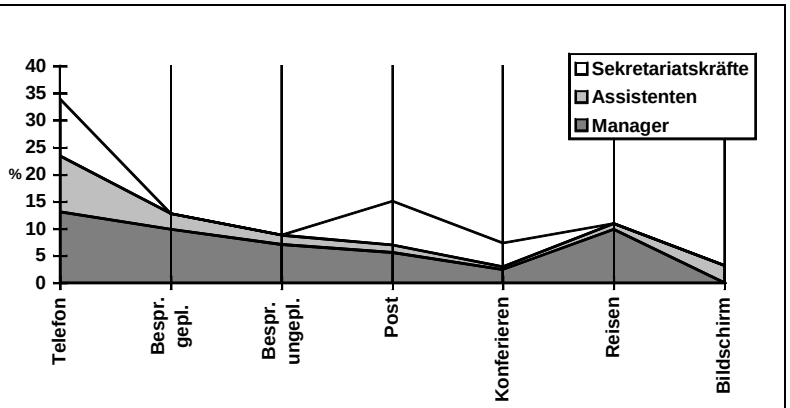


Abb. 2-4:	Verteilung der Kommunikationstätigkeiten auf Beschäftigungsgruppen

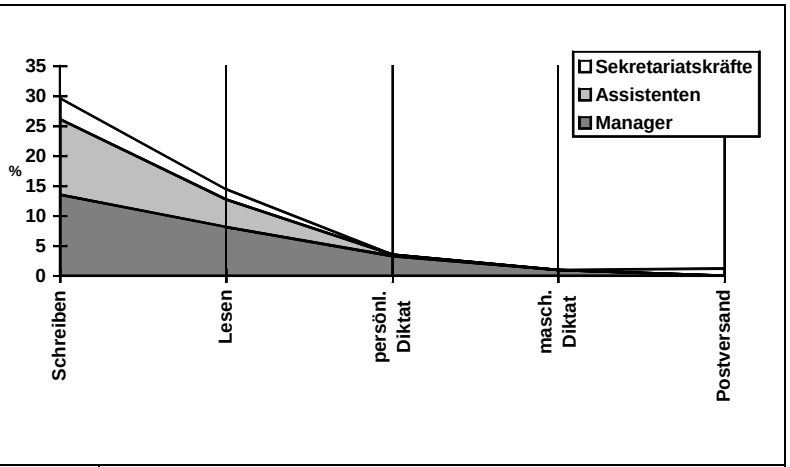


Abb. 2-5:	Verteilung der Schriftguthandhabungstätigkeiten auf Beschäftigungsgruppen

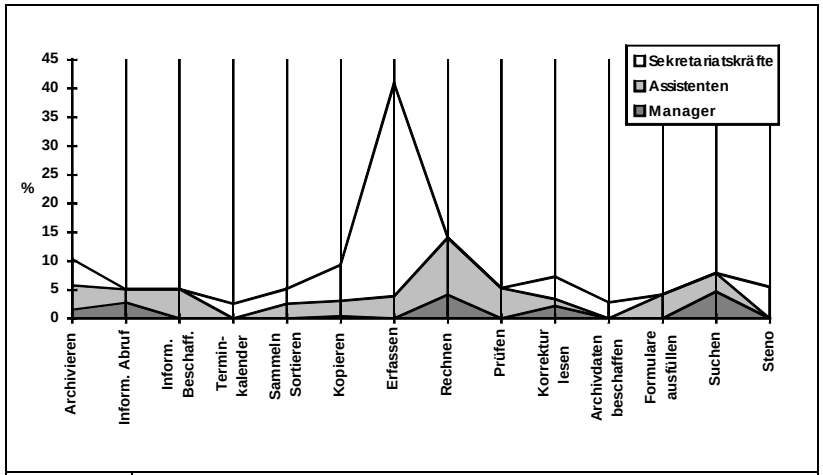


Abb. 2-6:	Verteilung der Informationsverarbeitungstätigkeiten auf Beschäftigungsgruppen

Sie beantworten die Frage, welche Tätigkeiten schwerpunktmäßig von welcher Beschäftigtengruppe ausgeführt werden und zeigen damit an, bei wem Auswirkungen technologischer Veränderungen primär wirksam werden. Dabei wurden einerseits Top- und mittleres Management zur Gruppe der Manager sowie Fachkräfte und Unterstützungspersonal andererseits zur Gruppe der Assistenten zusammengefaßt. Dieser Vorgehensweise liegt die Annahme zu Grunde, daß sie sich jeweils ähnlich gegenüber dem Einsatz neuer Technologien verhalten.

Da die bisher zitierten Studien überwiegend in den 60er Jahren und im anglikanischen Sprachraum durchgeführt wurden, sind Zweifel angebracht, ob ihre Ergebnisse sich auch für den europäischen bzw. deutschen Kulturkreis der 90er Jahre bestätigen lassen bzw. übertragbar sind. Schließlich haben sich das Aktionsfeld der Unternehmen durch Internationalisierung sowie das technische Instrumentarium wesentlich verändert.

Aufgrund der geringen Stichprobe zwar nicht repräsentative aber dennoch tendenziell verwertbare Antworten auf diese Fragen kann eine Replikationsstudie mittelständischer Unternehmen von Schreyögg/Hübl aus dem Jahre 1988 in Deutschland geben (vgl. [Schreyögg 91, Manager-Arbeit]). Die durchgängige Gegenüberstellung mit der "klassischen" Studie von Mintzberg 1968 in den USA (vgl. [Mintzberg 73, Managerial Work], S. 230 ff.), von Choran 1969 in Kanada (vgl. [Mintzberg 73, Managerial Work], S.

104 ff.) und von Kurke/Aldrich 1978 ebenfalls in den USA (vgl. [Kurke 83, Nature of managerial work]) bestätigt im wesentlichen deren frühe Ergebnisse (vgl. Abb. 2-7).

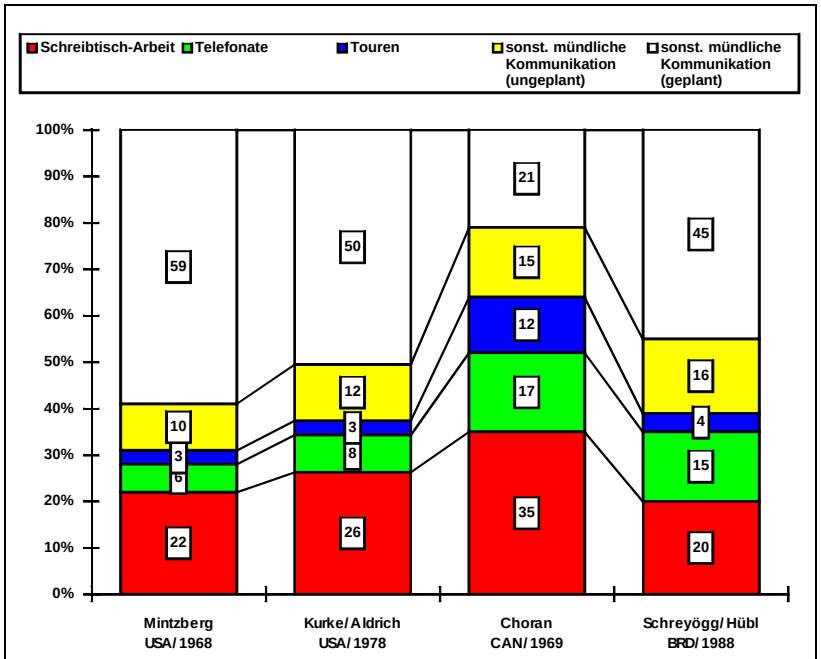


Abb. 2-7:	Zeitverteilung von Aktivitäten im Vergleich alternativer Studien (Quelle: [Schreyögg 91, Manager-Arbeit], S. 29)

Lediglich die Anteile für Telefonate und geplante, sonstige mündliche Kommunikation liegen zu Lasten der Schreibtischarbeit deutlich höher. Die Autoren relativieren letzteres durch die hohen Häufigkeiten der Aktivitäten, was unter dem Gesichtspunkt der Unterstützungspotentiale, deren Gestaltung und Wirkung zur Analyse der zeitlichen Dauer von Tätigkeiten überleitet.

2.2.2. Häufigkeiten und zeitliche Dauer von Tätigkeiten

Alle Studien weisen übereinstimmend eine hohe Anzahl Einzelaktivitäten pro Arbeitstag mit jeweils geringer Dauer aus (vgl. Abb. 2-8). So dominieren die Aktivitäten mit einer durchschnittlichen Dauer von weniger als 9 Minuten mit 78% gegenüber Tätigkeiten mit mehr als

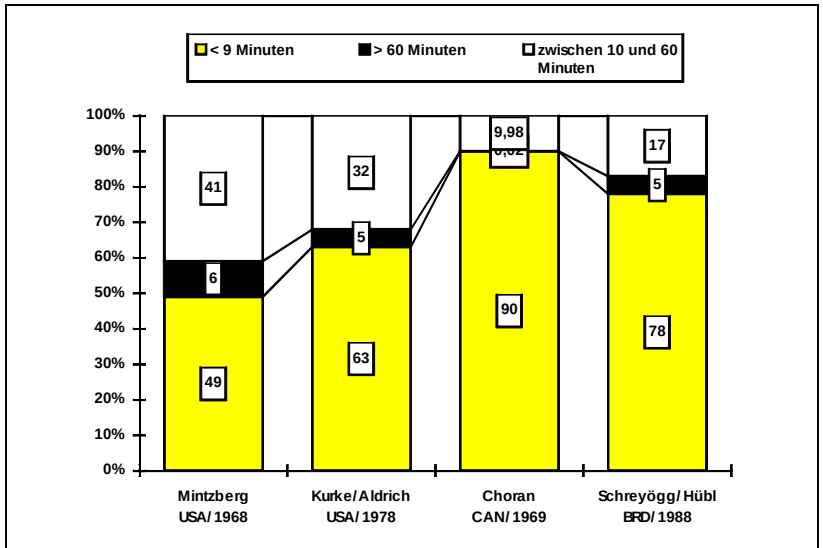


Abb. 2-8:	Dauer der Aktivitäten im Vergleich alternativer Studien (Quelle: [Schreyögg 91, Manager-Arbeit], S. 40)

1 Stunde von 5%. Eine tendenzielle Abschwächung dieses Effekts sehen Schreyögg/Hübl mit zunehmender Unternehmensgröße gegeben. Deutliche Differenzen ergeben sich (zwangsläufig) in Abhängigkeit mit der Tätigkeitsart. Im einzelnen konnten Schreyögg/Hübl 3 Minuten für Telefonate, 5 Minuten für ungeplante, sonstige mündliche Kommunikation und 10 Minuten für Schreibtischarbeit feststellen.

Die Bruchstückhaftigkeit von Führungstätigkeiten wird verstärkt durch die *"simultane Bearbeitung mehrerer Probleme und eine hohe Unterbrechungsrate bzw. Quote der Wiederaufnahme abgebrochener Sequenzen"* (vgl. [Schreyögg 91, Manager-Arbeit], S. 26). Die Vermutung fremdbestimmter, erzwungener Unterbrechungen, z.B. in Folge fehlender Informationen oder externer Anlässe, konnten die Autoren nicht bestätigen. Vielmehr kamen sie zu dem Ergebnis, daß die (befragten) Manager in 3/4 aller Fälle selbst den Wechsel zu anderen Aktivitäten einleiteten, um permanent in allen aktuellen Problemen "auf dem laufenden" zu sein, sowie diesen Arbeitsrhythmus durchaus beizubehalten wünschten.

2.2.3. Informationsquellen

Für die Beantwortung der Frage nach Unterstützungspotentialen sind neben Operationen und Häufigkeiten deren Inhalte, d.h. Objekte von entscheidender Bedeutung. Aufschlüsse

hierüber kann eine Studie von Jones und McLeod geben, die Informations-Transaktionen bezüglich Quelle, Medium und Bedeutung für Entscheidungsprozesse untersucht haben (vgl. [Jones 86, EIS], S. 220-249). Die Studie basiert auf freien Interviews mit Top-Managern von 5 US-Unternehmen sowie ergänzenden Fragebogen-Aktionen und Beobachtungen. Je Informations-Transaktion wurde zusätzlich eine subjektive Einschätzung der Wertigkeit der Information erhoben, um damit Rückschlüsse auf die Qualität der zugehörigen Quellen und Medien ziehen zu können. In Abb. 2-9 ist die Qualitätsverteilung der erhaltenen Informationen kumulativ dargestellt.

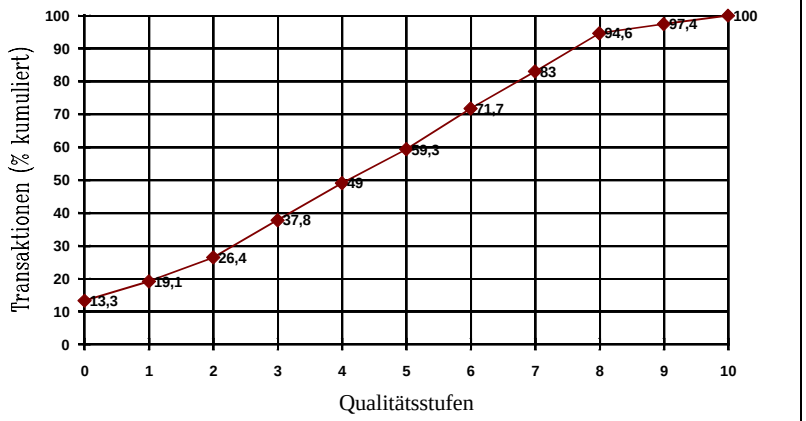


Abb. 2-9:	Qualitätsverteilung von Informations-Transaktionen (nach: [Jones 86, EIS], S. 229)

Mit über 25% bestätigt sich die häufig geäußerte Vermutung, daß Führungskräfte einer großen Menge unbedeutender oder geringwertiger Informationen ausgesetzt sind. Weniger als 10 % der Informations-Transaktionen erhielten hohe Werte. Die anschließenden Detail-Auswertungen nehmen jeweils Bezug auf diese Qualitätseinstufung.

Abb. 2-10 faßt die Resultate differenziert nach Informations-Quellen zusammen. Als Klassifizierungsmerkmal wurden sowohl intern-extern, als auch persönlich-anonym (Organisationseinheit) sowie die relative Stufe in der Organisationshierarchie berücksichtigt.

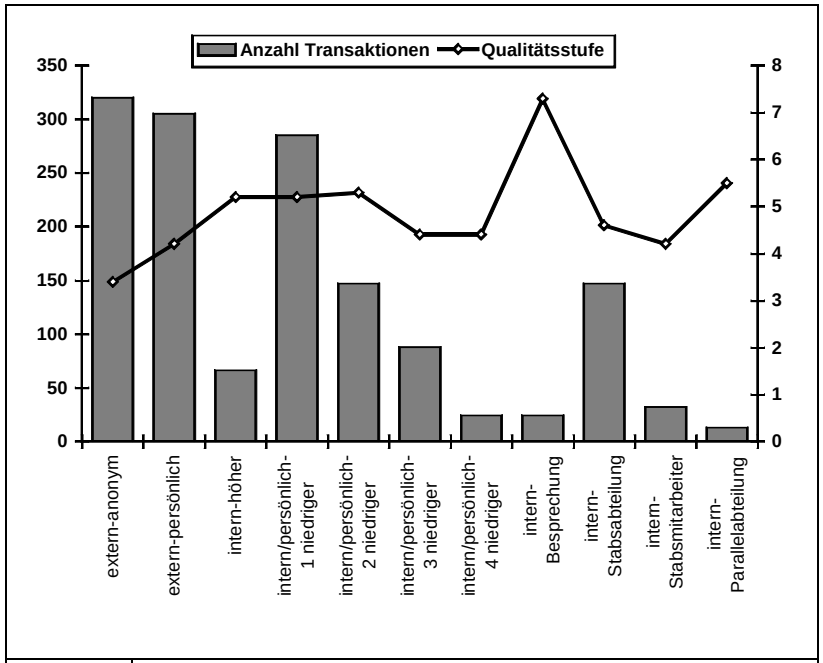


Abb. 2-10:	Informations-Transaktionen nach Quellen (nach: [Jones 86, EIS], S. 231)

Dabei zeigt sich, daß interne Quellen mengen- und qualitätsmäßig überwiegen und daß persönliche Quellen mit zunehmender Nähe in der Organisationshierarchie höher eingeschätzt werden.

Für die Analyse nach den Medien bei der Informations-Transaktion unterscheiden Jones/McLeod grob in verbale und schriftliche. Abb. 2-11 stellt die zugehörigen Ergebnisse in analoger Weise dar.

Daraus ist zu entnehmen, daß Telefonate und fast alle schriftlichen Medien zwar dominieren, keines von ihnen jedoch zu den höchst bewerteten gehört. An oberster Stelle stehen ausschließlich die übrigen verbalen Informationsformen.

Aus der Kombination von Quellen und Medien ziehen Jones/McLeod eine Reihe weiterer Schlüsse, von denen unter dem Gesichtspunkt der Rechnerunterstützung schwerpunktmäßig

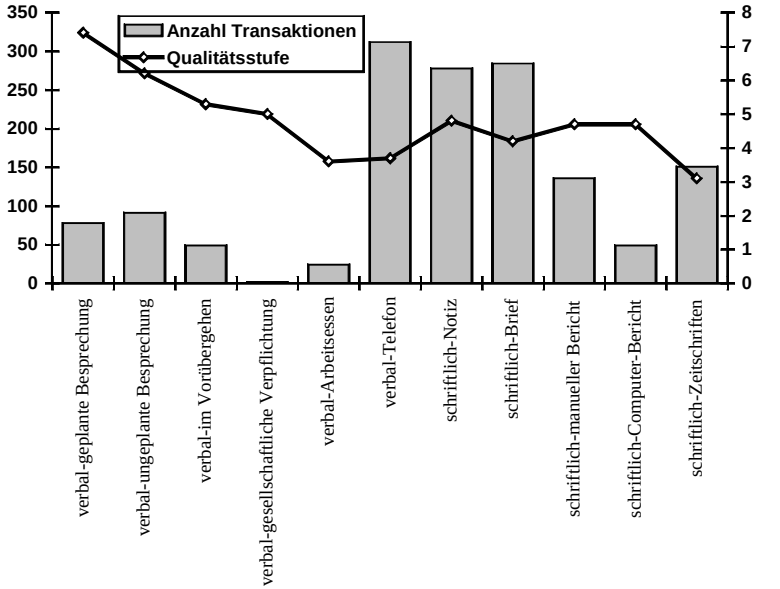


Abb. 2-11: Informations-Transaktionen nach Medien (nach: [Jones 86, EIS], S. 233)

die schriftlichen betrachtet werden sollen: (Kurz-)Notizen machen dabei nicht nur den größten Anteil aus, sie kommen in der Mehrzahl (60%) auch aus der direkten Umgebung der Manager und verfügen über eine hohe qualitative Einschätzung. Briefe, hauptsächlich von externer Natur (85%), erhielten dagegen eine relativ niedrige Bewertung. Nach Quellen unterschiedlich bewertet sind auch die formalen Berichte. Insgesamt im Durchschnitt niedrig eingestuft, ragt jedoch ein kleiner Teil - der aus Linienfunktionen - mit überdurchschnittlichen Qualitätseinstufungen heraus. Dabei wurde in ergänzenden Interviews zwar der potentielle Nutzen rechnergestützter Berichte eingeräumt, gleichzeitig jedoch der hohe Zeitaufwand für kontinuierliches Studium und Analyse derselben als Grund für die Abwertung herausgestellt.

2.3. Unterstützungspotentiale

Bei allen Vorbehalten, die Schlußfolgerungen auf der Basis der beschriebenen empirischen Untersuchungen aufgrund ihres Alters, Stichprobenumfanges und Kulturkreises entgegengebracht werden muß, können zusammenfassend jedoch folgende Feststellungen getroffen werden:

- Der Arbeitsplatz von Führungskräften ist durch eine Vielzahl von Einzelaktivitäten mit jeweils relativ kurzer Dauer gekennzeichnet, die zudem - fast in der Regel - mehrmals unterbrochen und später wieder aufgenommen werden.
- Dabei steht Kommunikation, speziell in verbaler Form im Vordergrund. Aufgrund des damit verbundenen Informationsreichtums (Gestik, Mimik, Tonfall, etc.) und Gestaltungsspielraums (Unterbrechen, Nachfragen, etc.) sowie der hohen sozialen Bedeutung (Menschenführung) ist die Bereitschaft zu Abstrichen als äußerst niedrig einzuschätzen. Das Fehlen einiger Komponenten des Informationsreichtums kann auch als Grund für die qualitative Abwertung der zahlreichen Telefonate interpretiert werden.
- Schriftliche und formale Informationsquellen werden zwar als potentiell nützlich eingestuft, ihre Menge und zeitaufwendige Handhabung, insbesondere im Hinblick auf die Selektion von in einem aktuellen Kontext benötigte Informationen oder beachtenswerte, weil andere Aktivitäten auslösende Ausnahmesituationen, läßt sie häufig in den Hintergrund treten.

Somit ergeben sich bei den Spitzen-Führungskräften nur wenige Ansatzpunkte für eine nutzbringende Rechnerunterstützung mit Akzeptanzchancen. Insbesondere muß davon ausgegangen werden, daß der absolute quantitative Nutzen (Zeitgewinn, etc.) relativ gering ausfallen wird, da die entsprechenden Informationsquellen und -medien derzeit bereits eine untergeordnete Rolle spielen. Dafür besteht die Chance - bei gleichbleibenden oder evtl. sogar reduziertem Zeitaufwand - die in übereinstimmender Einschätzung vermuteten Informationspotentiale bisher bereits rechnergestützt oder manuell erstellter Berichte zu erschließen und damit zu einer qualitativen Verbesserung des Informationsstands zu gelangen. In Frage kommende Leistungen hierfür sind also Reduktion der Informationsmenge durch flexible, schnelle Selektion relevanter Information.

Dies muß - und wird erwartungsgemäß - nicht zwangsweise (sofort) mit einer "Computerisierung" des Vorstands einhergehen. Reichwald spricht im Zusammenhang mit der Frage, ob die Führungskraft selbst oder über einen Mittelsmann von neuen rechnergestützten Hilfsmitteln Gebrauch machen soll, von Autarkie- und Kooperations-Konzept (vgl. [Reichwald 89,

Bürokommunikationstechnik], S. 11). Andere Autoren sprechen von Dialog und Trialog (vgl. [Lippold 82, Management], S. 290f.). Müller-Böling führt folgende Argumente an (vgl. [Müller-Böling 90, IuK-Anforderungen für Führungskräfte], S. 109):

❒ *Pro Dialog-Konzeption:*

- Unabhängigkeit von Assistenzkräften (vgl. [Tiemeyer 89, PC-Nutzung], S. 29; [Reichwald 89, Bürokommunikationstechnik], S. 12)
- ganzheitliche Aufgabenerfüllung
- Vorbildfunktion im Unternehmen
- realistische Einschätzungen über die tatsächlichen Möglichkeiten der Informations- und Kommunikationstechnik

❒ *Pro Trialog-Konzeption:*

- Hard- und Software erfordern zuviel Zeit für den ungeübten Manager (vgl. [Heilmann 87, Computerunterstützung], S. 13)
- umfangreiche Planungsprobleme sind nichts für den sporadischen Benutzer
- Navigation in Datenbanken sind zu aufwendig.

Aufgrund der eingangs geführten Rollen-Diskussion sowie des resultierenden Zeitgerüsts dürfte im ersten Schritt - aus beiderseitigen - Akzeptanzgründen eher das Unterstützungspersonal von Führungskräften den eigentlichen Anwender darstellen, um quasi seine "Dienstleistungs"-Qualität zu erhöhen. Für Führungskräfte selbst dürfte an erster Stelle aufgrund der in den Studien bestätigten hohen Bedeutung eine elektronische Notizverwaltung mit Funktionen der "Grundversorgung", wie Terminkalender oder Mail-Anschluß stehen; alles Dinge, die unter dem Schlagwort *persönliches Informations-Management* (PIM) subsummiert werden. Die nachfolgend diskutierten Konzepte und Werkzeuge müssen daher unter einem erweiterten Anwenderspektrum gesehen werden, das sich jedoch von vorneherein auf eine perspektivische Ausweitung in Führungsetagen einzustellen hat.

3. Das Konzept der Executive Information Systems (EIS)

3.1. Historische Entwicklung

Während Rechnerunterstützung auf operativen und dispositiven Unternehmensebenen mittlerweile zum selbstverständlichen Hilfsmittel avanciert ist, schienen die mittleren und besonders obersten Führungskräfte-Ebenen lange Zeit nicht dafür geeignet. Bekräftigt wurde diese These maßgeblich durch das Scheitern der Management-Informations-Systeme (MIS) Anfang der 70er Jahre. Im Rückblick ist dieses Scheitern zum größten Teil auf einen mangelhaften Entwicklungstand der seinerzeit zur Verfügung stehenden Hard- und Software zurückzuführen. Allerdings wurden in einer Art naiver Anfangseuphorie auch gravierende Fehler bei der Zielsetzung begangen, die zu einer praktisch unerfüllbaren Erwartungshaltung führten.

Diese Fehler wurden bereits zu Beginn der MIS-Ära von Ackoff in folgenden 5 Thesen beschrieben [Ackoff 67, MIS-Fehler]:

❒ *"give them more"*

Die Annahme, die Entscheidungsqualität durch quantitativ mehr Informationen verbessern zu können, führt stattdessen nur zu einer Überflutung mit zunehmend irrelevanter Information und behindert im Gegenteil eher die Entscheidungsfähigkeit und damit auch deren Qualität. Das zweifellos notwendige Mehr an Information muß demnach eher in Termini wie *schnell* und *relevant* interpretiert werden.

❒ *"the manager needs the information that he wants"*

Alle MIS-Ansätze gingen davon aus, die Führungskraft kenne im vorhinein ihren Informationsbedarf und könne diesen auch artikulieren. In der Realität erweisen sich jedoch zunächst als notwendig erachtete Informationen als unbedeutend, andererseits treten im Laufe von Entscheidungsprozessen immer wieder neue, zuvor nicht bedachte oder als unwesentlich angesehene Informationen hinzu. Umfang und Inhalte der bereitgestellten Information müssen sich daher dynamisch verändern und anpassen lassen.

❒ *"give a manager the information he needs and his decision making will improve"*

Eine allein durch den Faktor Information bestimmte Sichtweise der Qualität von Entscheidungsprozessen vernachlässigt sträflich die eigentlichen Mechanismen der Informationsverwendung und -verarbeitung, z.B. deren Beobachtung im Zeitverlauf und im Vergleich zu anderen Unternehmen, deren Zusammenführung usw.. Erst eine Ergänzung

mit handhabbaren Verwaltungs- und Verarbeitungsfunktionen bietet die Chance zu echter, und damit akzeptierter Unterstützung.

- *"more communication means better performance"*
  MIS zielten berechtigtermaßen auch auf eine Verbesserung des Informationsaustauschs im Unternehmen. Dies wurde jedoch zunächst wieder primär quantitativ und aus Sicht des MIS gesehen; insbesondere blieben soziale und organisatorische Aspekte außen vor, so daß ein Manager bei einem Wechsel zu rechnergestützter Informationsverwaltung und -verarbeitung auf seinen gewohnten und wohl auch benötigten individuellen Aktions- und Gestaltungsspielraum verzichten mußte. Ein MIS muß diesen sozialen und organisatorischen Aspekten durch entsprechende Zugriffsmechanismen und Verarbeitungsfunktionen Rechnung tragen, wenn es Akzeptanz erzielen will.

- *"a manager does not have to understand how an information system works, only how to use it"*
  Das Konzept, technische Hilfsmittel dem Anwender als Black-box anzubieten, ist weit verbreitet und entbehrt angesichts zunehmender Komplexität technischer Systeme nicht einer gewissen Logik. Nichtsdestotrotz gehen davon nicht zu vernachlässigende Gefahren in Bezug auf die Akzeptanz des Systems aus, insbesondere was Glaubwürdigkeit und Vertrauen in die Exaktheit und Gültigkeit des Systemoutput und dessen Verhalten angeht. Nicht umsonst spielen Erwartungskonformität, Beherrschbarkeit und Steuerbarkeit als Kriterien der Softwareergonomie eine zentrale Rolle. Ergebnisse von MIS müssen daher jederzeit für den Anwender nachvollziehbar, erklärbar und damit kontrollierbar sein.

Diese klassischen Fehler wurden zwar im Laufe der Zeit von Seiten der Wissenschaft, Systemanbietern und Entwicklern relativ schnell korrigiert, geblieben ist jedoch ein nicht ausrottbares Negativ-Image des Begriffs MIS. So kommt es, daß sich neuere Schlagworte mit nahezu identischer Zielsetzung und Definition am Markt größerer Beliebtheit erfreuen und durchzusetzen beginnen. Dies gilt insbesondere für den Begriff der Executive Information Systeme (EIS), für den im Deutschen die Bezeichnungen Führungs- ober Chef-Informationssysteme gebräuchlich sind.

3.2. Zielsetzung und Definition

Wenngleich noch keine allgemein anerkannte Definition für Führungsinformationssysteme existiert, läßt sich deren Zielsetzung an drei Schlüsselworten festmachen:

- *Führungsperson*
  Zunächst sind EIS auf eine bestimmte Zielgruppe von Anwendern gerichtet, die im folgenden als Führungspersonen bezeichnet werden sollen. Dieser als Äquivalent zum englischen *Executive* gewählte Begriff bezeichnet leitende Angestellte, die vornehmlich mit zwei Gruppen von Aufgaben betraut sind: zum einen der Kontrolle und Überwachung interner und externer Prozesse und Entwicklungen, zum anderen der Planung und Steuerung primär interner Aktionen.

- *führungs-relevante Information*
  Hauptaufgabe von EIS ist es, diese Zielgruppe mit führungs-relevanten Informationen zu versorgen. Was heißt führungs-relevant? Zunächst impliziert dies eine möglichst direkte und zeitnahe Bereitstellung von Informationen, da diese mit zunehmender Zahl zwischengeschalteter Instanzen bei der Informationsweitergabe und damit wachsendem Alter durch neuere Entwicklungen überholt werden sowie tendenziell als Basis für Beurteilungen und Entscheidungen an Bedeutung verlieren.

  Führungs-relevant impliziert weiterhin eine mengentheoretische Sicht von Information, d.h. nicht Einzelinformationen oder deren unstrukturierte Anhäufung bei noch so hoher Aktualität und Genauigkeit stehen im Vordergrund, sondern deren problemorientierte Zusammenfassung. Dies bedeutet in der Regel, daß Informationen aus verschiedensten Quellen, unternehmens-internen wie -externen, in Bezug auf einen oder mehrere Problemkontexte zusammengeführt werden müssen.

  Angesichts des speziellen Aufgabenspektrums der Zielgruppe, nämlich Kontroll- und Überwachungsaufgaben, heißt führungs-relevant auch, daß für diese Aufgaben typische Funktionen der Informationsverarbeitung bereits in die Informationsbereitstellung integriert sind. Somit sind führungs-relevante Informationen in der Regel abweichungs-orientiert in Form von absoluten oder prozentualen Veränderungen gegenüber Plan, Vorperiode oder auch Konkurrenz und allgemeiner Marktentwicklung. Hinzu treten Funktionen eines Ausnahme- oder Signalberichtswesens, bei dem gezielt Abweichungen außerhalb definierter Grenzwerte abgefragt werden können oder automatisch berichtet werden.

  Im Zusammenhang mit dem Kriterium direkter Versorgung erlauben EIS dem Anwender bei der Grenzwertdefinition eine aktive Rolle, so daß er selbst und jederzeit Art, Verknüpfung und Ausmaß der Informationsfilter bestimmen und an aktuelle Entwicklungen anpassen kann.

- *führungs-adäquate Form*
  Schließlich wollen EIS den Bedürfnissen und Restriktionen der besonderen Zielgruppe durch eine führungs-adäquate Form der Informationsversorgung Rechnung tragen. Hier-

zu gehört zuvorderst die Konzentration auf komprimierte und selektive Darstellung, um der Gefahr der Informationsüberflutung entgegen zu wirken. Mit der Komponente des Ausnahmeberichtswesens ist bereits ein wesentliches Mittel der Selektion angesprochen worden. Hinzu tritt die Bevorzugung aggregierter Informationen. Dabei ist allerdings darauf zu achten, daß Abweichungen, die sich im Zuge der Aggregation gegenseitig ausgleichen, nicht verloren gehen. Umgekehrt muß jederzeit die Möglichkeit erhalten bleiben, im Bedarfsfalle zu Detailinformationen "herabzuschreiten", um eine wirksame Abweichungsanalyse im Sinne der Lokalisation von Problemfeldern zu gestatten.

All dies muß in intuitiv verständlicher Form präsentiert werden bzw. bedienbar sein, da alle Untersuchungen (vgl. Kapitel 2.2) zeigen, daß bei der speziellen Zielgruppe immer nur von einem relativ geringen Prozentsatz der Arbeitszeit für die Nutzung eines Rechners ausgegangen werden muß. Einarbeitungszeiten oder gar der Rückgriff auf Bedienungshilfen, sei es in schriftlicher Form oder online, dürfen praktisch nicht entstehen.

Schließlich darf die Unterstützungsleistung des EIS sich nicht auf die Abfrage und Präsentation relevanter Informationen in konkreten Problemkontexten beschränken. Wesentlicher Bestandteil von Führungstätigkeit besteht in der Kommunikation gewonnener Erkenntnisse aus der Informationssammlung, -sichtung und -bewertung zu anderen Stellen und Arbeitsplätzen im Unternehmen. Diese Kommunikation kann unterteilt werden in reine Informationsweitergabe, zumeist verbunden mit der Erteilung von Arbeitsaufträgen, z.B. zur genaueren Klärung eines Sachverhalts, die Erarbeitung spezifizierter Zusatzinformationen bis hin zur Generierung alternativer Handlungsvorschläge und deren Folgen- bzw. Wirkungs-Abschätzung. Die Ergebnisse dieser Arbeitsaufträge sollen im Rahmen von EIS dann ebenfalls auf elektronischem Wege an den Auftraggeber zurückgemeldet werden. Dabei ist es wichtig, diese wieder im ursprünglichen, den Handlungsauftrag auslösenden Informationskontext zu erhalten bzw. in diesen einbinden zu können.

3.3. Abgrenzung und Einordnung

Aus der beschriebenen Zielsetzung und Definition von EIS folgt unmittelbar, daß EIS nicht - wie anfänglich vor allem in dem Ursprungsland USA geschehen - als isoliertes Hilfsmittel und ausschließlich für Führungskräfte der obersten Unternehmensebene gesehen werden können. Aussicht auf Akzeptanz im Unternehmen kann nur bei einer vertikalen Durchdringung der Unternehmensorganisation bestehen (vgl. Abb. 3-1).

Während auf höheren Unternehmensebenen EIS-Komponenten den rechnerunterstützten

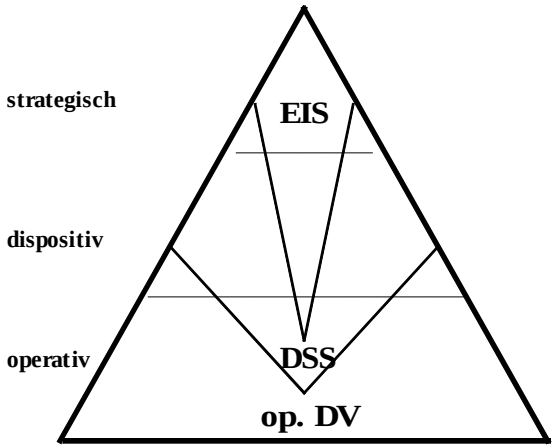


Abb. 3-1: EIS-Einordnung in die Unternehmenspyramide

Arbeitsplatz dominieren, kommt ihnen auf dispositiver Ebene eher ein kommunikativer Charakter im Sinne der Entgegennahme von Arbeitsaufträgen und der Aufbereitung weiterzuleitender Informationen zu. Auf dispositiver Ebene stehen dagegen modellbasierte Analyse- und Entscheidungsvorbereitungs-Funktionen im Vordergrund, für die eher Werkzeuge des Decision Support (DSS) in Frage kommen.

Gleichzeitig verändert sich entlang der Unternehmenshierarchie auch die Informationsbreite von EIS-Komponenten. Je mehr man sich der operativen Ebene nähert, desto detaillierter wird der Informationsbedarf und desto enger wird das Spektrum "angezapfter" Quellen. Beispielsweise verteilt sich eine aggregierte Kennzahl der obersten Ebene auf der nächst niedrigeren Ebene auf absolute Kenngrößen verschiedener Kostenstellen. Sie wird somit aus mehreren, tieferliegenden EIS-Komponenten gespeist, die ihrerseits in der Organisationshierarchie horizontal anzusiedeln sind.

3.4. Leistungsprofil

3.4.1. Klassifizierung des Leistungsprofils

Entsprechend der Zielsetzung rechnergestützter Arbeitsplätze für Führungspersonen ergibt sich ein typisches Profil an Leistungsanforderungen. Diese Anforderungen können in drei Klassen gruppiert werden, wie Abb. 3-2 verdeutlicht.

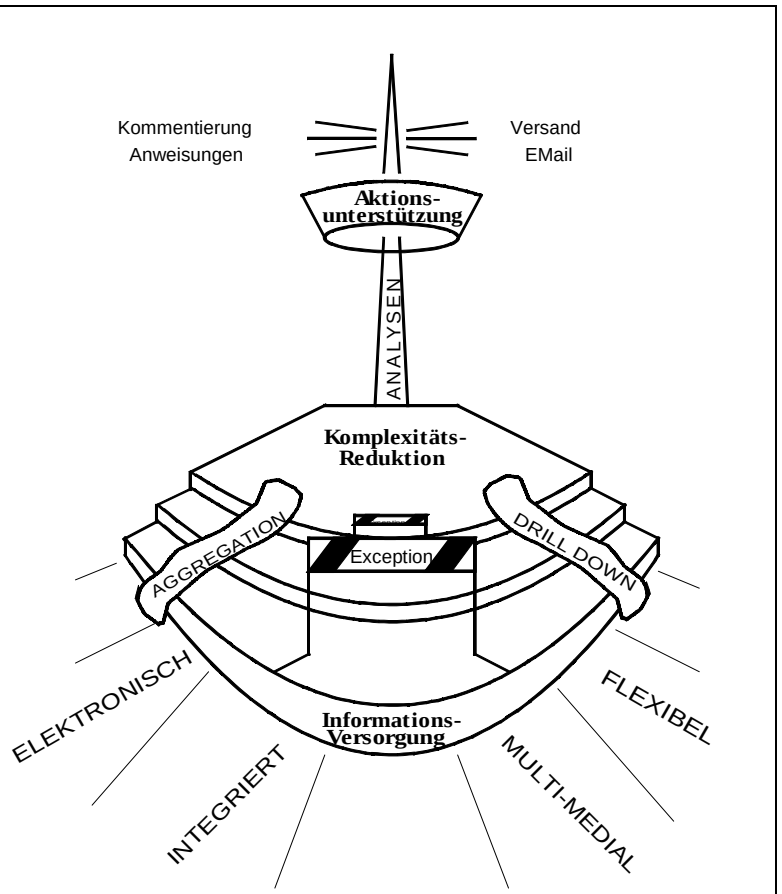


Abb. 3-2: EIS-Leistungsklassen

Es sind dies

- ❒ *Informationsversorgung*
- ❒ *Komplexitätsreduktion und*
- ❒ *Aktionsunterstützung.*

Bevor auf typische Funktionen der einzelnen Leistungsgruppen eingegangen wird, soll zunächst deren Zusammenwirken beschrieben werden.

Die *Informationsversorgung* stellt die elektronische Schnittstelle zu den verschiedenen Informationsquellen dar. Ihr kommt eine integrative Funktion in zweierlei Hinsicht zu:

- ❒ zum einen müssen unterschiedliche Formate und Abfragesyntax verschiedener interner und externer Quellsysteme homogenisiert werden,
- ❒ zum anderen müssen auch nicht ex ante in elektronischer Form vorliegende Informationen mit erfaßt werden können.

Daraus ergibt sich zwangsläufig die Forderung, multi-mediale Informationseinheiten (Objekte) verwalten und bearbeiten zu können, z.B. Texte verschiedener Textverarbeitungssysteme, alpha-numerische Abfrageergebnisse unterschiedlichster Daten(bank)-Verwaltungssysteme, Graphikformate verschiedener Graphiksysteme, bis hin zu Zeichnungen von CAD-Systemen, gescannten Bildern oder digitalisierter Sprachinformation. Dabei müssen die Informationsobjekte in ihrer Herkunft und Zusammensetzung jederzeit schnell und individuell den sich permanent ändernden Informationsbedürfnissen der Führungskräfte angepaßt werden können, also in hohem Maße flexibel sein.

Auf dieser vereinheitlichten, weil elektronisch erfaßten und weiter verarbeitbaren Informationsbasis setzt die zweite Klasse von Leistungen, die *Komplexitätsreduktion* auf. Ihre Aufgabe besteht darin, die Informationsmenge nach problembezogenen Kriterien zu filtern, zu verdichten und zu gruppieren. Dabei dürfen die Beziehungen zu den Basisinformationen nicht verlorengehen, damit im Bedarfsfall schnell auf sie zurückgegriffen werden kann.

Als letzte Klasse von EIS-Leistungen schließt sich die Unterstützung typischer Führungsaktionen im Zusammenhang mit der Informationsverarbeitung an. Hierzu gehört zunächst der gesamte Bereich der privaten Organisation und Ablage von Informationen, d.h. die individuelle Sortierung und Kommentierung, insbesondere das Verknüpfen mit Wiedervorlageterminen oder -ereignissen. Es handelt sich dabei meist um Aktionen, die ohne EIS in Form von handschriftlichen Notizen oder per mündlicher Anweisung an das Sekretariat delegiert werden.

Das gleiche gilt für eine zweite, nach außen gerichtete Gruppe von Aktionen. Diese betreffen die Weitergabe originärer, d.h. selbst erstellter oder sekundärer, d.h. z.B. kommentierter oder um Handlungsanweisungen ergänzter Informationen an andere Mitarbeiter. Um Medienbrüche zu vermeiden, d.h. elektronisch abgefragte und am Bildschirm präsentierte Informationen nicht erst ausdrucken und dann mit den handschriftlichen Informationen wieder eingeben oder per Hauspost verschicken zu müssen, gehört zu einem EIS auch ein leistungsfähiges Electronic-Mail-System, das in der Lage ist, alle genannten Formate von Informationsobjekten zu versenden.

3.4.2. Funktionen der Leistungsklassen

3.4.2.1. Informationsversorgung

Um dem Leistungsspektrum der Informationsversorgung gerecht zu werden, ist eine Reihe typischer Funktionen notwendig. Da diese unabhängig von den konkreten Informationsobjekten vorhanden sein müssen, bietet es sich an, diese als vorgefertigte Konstrukte in einem Werkzeug zur Erstellung konkreter EIS-Anwendungen, den sogenannten EIS-Generatoren (s. Kapitel 5), bereitzustellen.

Für die Versorgung mit externen Informationen bieten sich eine Reihe externer Informationsdienste an. Beispiele hierfür sind

- ❒ *DowJones News Retrieval*
- ❒ *CompuServe*
- ❒ *Reuters*
- ❒ *DIW*
- ❒ *Statistisches Bundesamt etc.*

Da alle diese Dienste eigene Datenformate und Kommunikationsprotokolle besitzen, sind geeignete Schnittstellenkomponenten zu entwickeln, die das heterogene Informationsangebot in eine einheitliche Abfrage- und Präsentationsoberfläche transformieren. Dazu wird im Regelfall eine eigene Zwischenspeicherung und Verwaltung abgefragter Informationsblöcke gehören, um aufwendige Leitungskosten zu sparen, sofern der Informationsanbieter dies vertraglich gestattet.

Um zu einer einheitlichen Abfragesprache auf unterschiedliche, interne und/oder externe Datenbasen zu gelangen, bieten sich mehrere Lösungsalternativen an. Zum einen kann eine übergeordnete, mächtige Abfragesprache, z.B. auf SQL-Basis installiert werden, die über

Sprachkonverter direkt Verbindungen zu den jeweiligen Daten(bank)verwaltungssystemen herstellt. Hierbei sind jedoch zwei Nachteile zu konstatieren:

- ❒ erstens kann aufgrund differierender Datenstrukturen in den angeschlossenen Datenspeichern, z.B. hierarchische vs. relationale Datenmodelle, nicht immer der volle Funktionsumfang der Meta-Abfragesprache realisiert werden;
- ❒ zweitens ist bei diesem Direktanschluß an originäre und damit operative Datenbasen mit langen Bearbeitungszeiten sowie einer inakzeptablen Störung des operativen Geschäftsbetriebs zu rechnen.

Hinzu kommt, daß in den seltensten Fällen ein derart hohes Maß an Aktualität erforderlich ist, im Gegenteil dürfte dies in der Regel sogar stören. Daher bietet sich ein zweistufiges Vorgehen bei der Informationsversorgung an, das die Installation einer spezifischen EIS-Datenbasis vorsieht:

- ❒ im ersten Schritt werden mittels Batchläufen zu festgesetzten Zeitpunkten, z.B. täglich, wöchentlich oder monatlich potentiell relevante Daten aus den operativen Systemen in die EIS-Datenbasis eingespeist. Dabei können neben der notwendigen Validierung vor allem Restrukturierungen in Bezug auf verknüpfte Abfragen sowie vorzeitig zeitaufwendige Berechnungen, z.B. zur Aggregation durchgeführt werden.
- ❒ Im zweiten Schritt können dann konkrete EIS-Anwendungen, bei Bedarf auch adhoc über eine einheitliche Abfrageschnittstelle, die zudem graphisch aufbereitet und leicht bedienbar gestaltet werden kann, auf diese EIS-Datenbasis zugreifen. Die Verteilung dieser EIS-Datenbasis auf verschiedene physikalische Ressourcen bzw. logische Untereinheiten bleibt davon unberührt.

Im Interesse der Flexibilität ist die Informations-Versorgung und -Präsentation von EIS-Anwendungen *datengetrieben* zu realisieren. Datengetrieben bedeutet dabei, daß Änderungen der Datenbasis ohne korrigierende oder ergänzende Eingriffe des EIS-Entwicklungspersonals oder -Anwenders von der EIS-Anwendung bzw. -Oberfläche quasi automatisch berücksichtigt werden. Da Änderungen der Datenbasis vielfältigen Charakter aufweisen können, muß dieses Kriterium sehr differenziert betrachtet werden.

Im Trivialfall bedeutet datengetrieben, daß numerische Änderungen sich direkt, z.B. in Form veränderter Graphiken, niederschlagen. Komplizierter wird es jedoch, wenn strukturelle Änderungen der Datenbasis hinzukommen, im einfachsten Fall, daß statt bisher n Ländern nunmehr n+m Ländereintragungen existieren und darzustellen sind. Neben der erwünschten automatischen Erweiterung von tabellarischen oder graphischen Darstellungen, bis hin zur

automatischen Einfügung eines Rollbalkens für den Tabellen- oder Graphikbereich, treten Umsetzungsprobleme insbesondere im Zusammenhang mit weiterführenden Funktionen der graphischen Benutzeroberflächen auf. Im Falle einer vorgesehenen Weiterverzweigung des Informationsabrufs über die aus der Datenbasis abgerufenen und tabellarisch dargestellten Länder müssen nämlich nunmehr n+m Aktionsfelder generiert werden. Gerade angesichts der hohen Dynamik von EIS-Datenbasen in dieser Hinsicht kommt einer tiefen Implementierung des Kriteriums datengetrieben immense wirtschaftliche Bedeutung zu.

3.4.2.2. Komplexitätsreduktion

Auch und gerade im Bereich der Komplexitätsreduktion lassen sich eine Reihe typischer, allgemein verwendbarer Funktionen bestimmen, die für die Implementierung als generische Konstrukte von EIS-Generatoren in Frage kommen.

An erster Stelle ist im Zusammenhang mit der Konzentration auf die Darstellung aggregierter Daten die Funktion des *Drill-Down* zu nennen. Darunter wird eine Kombination aus Funktionalität und Bedienung verstanden. Funktional bedeutet dies den impliziten Abruf von Detailinformationen zu einer am Bildschirm dargestellten Informationseinheit, z.B. einem Jahressummenwert oder einer aggregierten Kennzahl. Bedienungsmäßig wird diese Funktion direkt durch das Informationsobjekt selbst ausgelöst, d.h. bspw. durch Mausklicken auf das Objekt, sei es nun eine Zahl, ein Bild, ein Text oder auch eine Graphik bzw. ein Graphikteil.

Im Regelfall wird unter Detailinformation eine Information gleichen Typs auf niedrigerer Aggregationsstufe verstanden, also im o.g. Beispiel etwa die Monatswerte. Da zumeist aber Disaggregationen nach verschiedenen Dimensionen möglich sind, z.B. bezüglich Produkten, Regionen oder der Zeit, kann dieser Vorzug intuitiv leichter Bedienbarkeit in der Praxis leicht und schnell zu Verwirrung und Unsicherheit auf Seiten des Anwenders führen. Einen Ausweg bietet die Zwischenschaltung eines Popup-Menüs, in dem die alternativen Detaillierungsrichtungen zur Auswahl angeboten werden, um den Preis eines weiteren Interaktionsschrittes. Dennoch bietet diese Alternative einen weiteren Vorteil: außer der Abfrage von Detailinformationen gleichen Typs können auf diese Weise nämlich auch Zusatzinformationen jedweder Art integriert werden, z.B. textuelle Kommentare, ergänzende Bildinformationen usw..

Eine weitere häufig in EIS-Generatoren verfolgte Strategie besteht in der Kombination von Markierung und fest in die Oberfläche installierten, ikonenhaft repräsentierten Werkzeugen, z.B. Ikonen für die verschiedenen Disaggregationsdimensionen, für Kommentare, definitorische Erklärungen usw.. Diese können dann je nach Abrufmöglichkeiten des markierten Informationsobjekts aktiv oder inaktiv, d.h. abgedunkelt, dargestellt werden.

Als Problem bleibt jedoch in jedem Fall, dem Anwender bei einer Vielzahl selektierbarer Informationsobjekte visuell zu signalisieren, welche Objekte über welche Zusatzinformationen verfügen. Eine nur partiell brauchbare Lösung in dieser Hinsicht stellt ein Ansatz dar, die Werkzeugikonen direkt mit den jeweiligen Informationsobjekten in Form eines Fensters zu verbinden, da hierdurch einerseits das gleichzeitig darstellbare Informationsvolumen stark eingeschränkt und andererseits die Übersichtlichkeit, z.B. bei Zeitreihenvergleichen, verlorengeht. Daraus folgt, daß je nach darzustellender Informationsart zu entscheiden ist und daher möglichst verschiedene Realisierungen für das Drill-Down in EIS-Generatoren zu implementieren sind.

Die zweite wesentliche Funktionskomponente zur Komplexitätsreduktion ist in einem leistungsfähigen *Exception-Reporting* zu sehen. Ähnlich wie schon beim Kriterium datengetrieben ist hier eine sehr differenzierte Betrachtung notwendig. Um überhaupt einen Beitrag zur Komplexitätsreduktion leisten zu können, genügt es keinesfalls, lediglich Abweichungen von vorgegebenen Toleranzbereichen visuell zu markieren, vielmehr ist damit eine Selektionsfunktion zu verbinden. Da Abweichungen an den verschiedensten Stellen der Datenbasis, insbesondere auf verschiedenen Aggregationsstufen auftreten können und ein EIS seine Informationen in wohlorganisierter Form, d.h. in der Regel als Baum, Netz oder Sequenz wohldefinierter und formatierter Bildschirm(fenster)einheiten präsentieren muß, bietet sich für die Implementierung eines selektiven Exception-Reporting folgende Lösung an: alle Abweichungsobjekte werden mit Definition oder Bezeichnung, aktuellem Wert, Toleranzwerten sowie Informationen darüber, in welchen, zu diesem Zwecke ebenfalls benannten Bildschirmen sie vorkommen, in einem eigenen Datenbasisbereich gesammelt. Über ein spezifisches Exception-Werkzeug, eine eigene Sequenz von EIS-Bildschirmen und -Funktionen, kann es jederzeit von jedem Ort der EIS-Anwendung aufgerufen werden und nach verschiedenen Kriterien sortiert und gruppiert Abweichungsinformationen anbieten, z.B. nach Aggregationsstufen oder nach Bildschirmen. Bei konsequenter Integration des Drill-Down-Prinzips (s.o.) kann dann unmittelbar sowohl von aggregierten Abweichungen zu Detailabweichungen als auch zu einem der Bildschirme gesprungen werden, in dem diese Abweichung in ihrem Kontext dargestellt ist. Die Anwendung des Drill-Down-Prinzips auf die Aggregationsstufen des Exception-Reporting erfordert dabei eine weitere Funktionalität: Abweichungen einer niedrigeren Ebene, die sich auf nächsthöherer Ebene gegenseitig kompensieren, müssen dort trotz Einhaltung der Toleranzgrenzen entsprechend gekennzeichnet angezeigt werden.

Auf diese und weitere Aspekte des Exception-Reporting wird im Detail unter Punkt 5.3.5 eingegangen. Sie sollen hier nur aus Gründen der Vollständigkeit genannt werden:

- *Margenspektrum*
- *Margenhierarchie*
- *Ausnahmeobjekttypen*

Schließlich impliziert ein leistungsfähiges Exception-Reporting eine automatische Information des Anwenders über Umfang, Grad und Ursachen von Ausnahmen. Dies kann durch ereignisorientiert ablaufende Funktionen, sogenannte Triggerfunktionen, realisiert werden. Da sich ein automatisches Aufblättern aller Einzelabweichungen bei großen Abweichungsmengen in der Regel störend auswirken wird, bietet sich eine Beschränkung auf einen summarischen optischen Hinweis bei Systemstart mit Verzweigungsmöglichkeit in die Exception-Funktion an.

In diesem Zusammenhang soll auf ein weiteres sinnvolles Einsatzgebiet für Triggerfunktionen hingewiesen werden, das als prospektive Disaggregation bezeichnet werden soll: darunter soll verstanden werden, daß die Tiefe des abrufbaren Informationsangebots in Abhängigkeit der aktuell vorliegenden Abweichungen gesteuert wird, d.h. falls etwa Kennzahlen einer bestimmten Region außerhalb des definierten Toleranzbereichs liegen, wird automatisch die Versorgung der EIS-Datenbasis über das normale Maß hinaus um weitere Detaillierungsstufen erweitert.

3.4.2.3. Aktionsunterstützung

Zu den generischen Konstrukten im Bereich der Aktionsunterstützung eines EIS-Generators gehört zunächst die Kommentierungsmöglichkeit jedes beliebigen Informationsobjekts durch den Anwender. Als kommentierfähiges Objekt kommen dabei neben ganzen Bildschirmeinheiten insbesondere auch einzelne Daten oder Datengruppen in Betracht. Dabei muß sowohl dem Anwender die Möglichkeit gegeben werden, eigene Datengruppen zu definieren, als auch sichergestellt sein, daß hinterlegte Kommentare wahlweise automatisch auch in anderen Bildschirmeinheiten, in denen die gleichen Daten oder Datengruppen auftreten, automatisch wieder zur Verfügung stehen.

Weiterhin ist durch ein integriertes, leistungsfähiges EMail-System der Versand beliebiger EIS-Inhalte an beliebige Teilmengen von Mitarbeitern ohne Medienbruch zu gewährleisten. Wahlweise sollte der EIS-Generator über Schnittstellen zu marktgängigen EMail-Systemen verfügen.

Ein Sonderfall des Versands stellt die Wiedervorlage zu bestimmten Terminen oder Ereignissen, z.B. Sitzungen, dar. In diesem Zusammenhang kann das EIS zum kombinierten persönlichen Kalender und Notizbuch entwickelt werden. Von besonderem Nutzen in diesem Zusammenhang kann es sein, Eintragungen des EIS im Notizbuch dynamisch mit der Datenbasis zu verbinden. Das hieße dann bspw., daß für die monatlichen Sitzungen regelmäßig benötigte Informationseinheiten automatisch in aktueller Form bereitgestellt werden können und bei entsprechender, integrierter Archivierungsfunktion darüber hinaus auf Vormonatsberichte zurückgegriffen werden kann. In einer weiteren Ausbaustufe sind ferner analytische EIS-Bildschirme zum Vergleich der Bildschirminhalte der archivierten Historie denkbar. Somit reichen die Funktionen zur Aktionsunterstützung durchaus in den Bereich des Decision Support hinein.

3.5. Kritische Erfolgsfaktoren (CSF) für EIS

3.5.1. Übersicht und Klassifizierung der CSF

Von Theorie und Praxis werden eine Vielzahl von Faktoren beschrieben, die bei Entwicklung und Einführung von EIS in Unternehmen als erfolgskritisch für diese rechnergestützten Systeme angesehen werden. Zumeist basieren sie auf persönlichen Erfahrungen konkreter Einführungsprojekte in Unternehmen oder empirischen Untersuchungen darüber (vgl. [Rockart 88, Executive Support Systems], S. 151ff.). Die beschriebenen Faktoren lassen sich zwei großen Gruppen zuordnen: organisatorische und technische (vgl. Abb. 3-3).

Die *organisatorischen Erfolgsfaktoren* beziehen sich auf den Kreis der an der Einführung mittelbar oder unmittelbar Beteiligten, insbesondere deren fachlicher Qualifikation, Rolle und Kompetenz, Eigenschschaften des zu realisierenden Einstiegsproblems sowie zu schaffende Infrastruktur-Voraussetzungen in Bezug auf Aufbau- und Ablauf-Organisation. Auf ihre Nichtberücksichtigung wird überwiegend das Scheitern von EIS-Realisierungen zurückgeführt.

Die *technischen Erfolgsfaktoren* betreffen Hard- und Software der EIS-Realisierung sowie eingesetzte Methoden und Vorgehensweisen der Entwicklung. Ihnen wurde nach allgemeiner Überzeugung bislang fälschlicherweise alleinige Aufmerksamkeit geschenkt. Jedoch geschah dies zumeist auf oberflächlichem Niveau in des Wortes reinster Bedeutung, d.h. im Mittelpunkt der Diskussion standen Elemente der graphischen Oberfläche, Integrationsmöglichkeiten verschiedenster Grafikformate usw.. Vernachlässigt wurden häufig Aspekte der Entwicklung, insbesondere der Weiterentwicklung und Wartung entwickelter Proto-

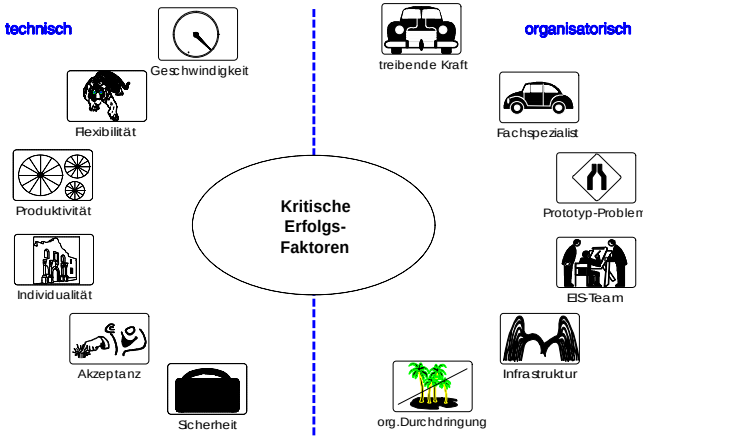


Abb. 3-3: Übersicht EIS-Erfolgskriterien

typen. Somit besteht gerade auch in diesem Bereich - wie die nachfolgende Detailanalyse zeigen wird - erheblicher Nachholbedarf.

3.5.2. Organisatorische Erfolgsfaktoren

3.5.2.1. Treibende Kraft (Executive Sponsor)

Rechnergestützte Führungsinformationssysteme implizieren Veränderungen der Verantwortlichkeiten und Wege der Informationsverteilung in der Unternehmenshierarchie (vgl. [Rockart 88, Executive Support Systems], S. 154). Historisch gewachsene, stellenmäßig organisierte Aufgaben der Sammlung, Zusammenstellung und Aufbereitung werden ganz oder teilweise ersetzt. Dies geschieht einerseits durch Automatisierung, andererseits dadurch, daß Informationen direkt via Terminal von der Führungskraft abgerufen werden können, ohne dazu Informationsmittler beauftragen oder einschalten zu müssen.

Damit dies funktionieren kann, müssen bisherige Informationsmittler für diese Reorganisation ihrer Rolle "gewonnen" werden. Wenngleich die erwünschte Straffung der Informationswege im Zuge von stellenmäßigen Zusammenfassungen von Aufgaben partiell zu Freisetzungen und damit zu einer tendenziellen Verflachung der Unternehmenshierarchie beiträgt, kommt einem anderen Aspekt größere Bedeutung zu: Informationsmittler müssen bereit sein, ihr Know how explizit zu machen und in rechnergestützte Abläufe zu transferieren. Dies wird in der Literatur häufig als potentieller Machtverlust dargestellt (vgl. [Rockart 88, Executive Support Systems], S. 154). Dabei wird jedoch übersehen, daß eine einmalige Preisgabe ihres Wissens zum Zwecke der Automatisierung keineswegs ausreicht. Vielmehr ist ihre permanente Einbindung zu kontinuierlichen Kontroll-, Kommentierungs- und Korrekturaufgaben notwendig, um Korrektheit, Gültigkeit und damit Akzeptanz der jetzt direkt abgerufenen Informationen seitens der Führungskräfte sicher zu stellen. Was sich für Informationsmittler ändert, sind also zum einen die eingesetzten Werkzeuge und zum anderen der Grad der Verflechtung mit anderen Informationsmittlern und der (zusätzlichen) Einsehbarkeit der übermittelten Informationen durch diese.

Zur Durchsetzung dieser gravierenden organisatorischen Änderungen ist es notwendig, daß EIS-Projekte von höchster Unternehmensebene initiiert, betrieben und durchgesetzt werden. Rockart faßt dies unter dem Begriff des "Executive Sponsor" zusammen (vgl. [Rockart 88, Executive Support Systems], S. 155). Ihm kommt grundsätzlich auch die Aufgabe zu, festzulegen, zu bestimmen bzw. zu fordern, welche Informationen über das EIS transportiert werden sollen.

3.5.2.2. Fachspezialist (Operating Sponsor)

Aus Zeitgründen und wegen der notwendigen detaillierten Sachkenntnis wird diese Rolle des inhaltlichen Ansprechpartners in der Regel an Fachspezialisten delegiert, deren Identifikation mit dem Projekt in gleichem Maße notwendig ist und von denen Rockart deshalb von sogenannten "Operating Sponsors" spricht (vgl. [Rockart 88, Executive Support Systems], S. 163). Dies sind Mitarbeiter oder Vertraute im unmittelbaren Umfeld, die traditionell Informationsbedarf, Arbeitsweise und -stil der Führungskraft bestens kennen, z.B. Assistenten oder persönliche Referenten des Vorstands oder der Geschäftsführung, aber auch Chef-Controller oder Funktionsbereichsleiter, also leitende Mitarbeiter der ersten Ebene unter der Geschäftsleitung.

Als Repräsentant der Führungskraft fungieren sie permanent über die gesamte Projektlaufzeit als Ansprechpartner und Vorentscheidungsinstanz für Inhalte und Form des EIS. In vielen Fällen werden sie zum eigentlichen, unmittelbaren Hauptnutzer und Anwender des realisierten Systems, insbesondere bei größerem Funktionsumfang des EIS in dem Sinne, daß nicht nur starr einmal festgelegte Informationsinhalte abgerufen werden können, sondern flexibel neue Informationsbedürfnisse zusammengestellt und aufbereitet werden. EIS verwandeln sich damit zu einem Produktivitätssteigernden Werkzeug in der Hand der unmittel-

baren Informationslieferanten; die Form der Kommunikation und Zusammenarbeit zwischen Führungskraft und Referenten bleibt dann weitestgehend erhalten.

Bei konsequenter Übertragung dieses Prinzips auf die nächst niedrigeren Hierarchieebenen wird die Akzeptanzproblematik durch potentiellen Machtverlust reduziert. Allerdings besteht die Gefahr der Zementierung historisch gewachsener Machtpotentiale durch rechnergestützte Abläufe. Eine Freigabe des Datenzugriffs über bisherige Abteilungs- oder Stellengrenzen hinaus ist daher in jedem Falle sicher zu stellen, während Kompetenzen zu wertmäßigen Prüfungen und Korrekturen bzw. bedeutungsmäßigen Kommentaren und Einschätzungen durchaus dort verbleiben, einbezogen und genutzt werden sollten.

3.5.2.3. Prototyp-Problem

Einführung, Durchsetzung und Verbreitung von EIS in Unternehmen ist ein Prozess von höchster Eigendynamik, im positiven wie im negativen Sinn: Erfolg und Akzeptanz erster Prototypen in einem Bereich produzieren Nachzieheffekte in anderen Bereichen, die anfänglich abweisende Haltungen und Bedenken aufgrund des realisierten zeitlichen und qualitativen Informations-Backlogs schnell zur Seite schieben. Um diese Eigendynamik in Gang zu setzen, d.h. einen tatsächlichen, spürbaren Informationsvorsprung zu realiseren, kommt der richtigen Wahl des Einstiegs- oder Prototyp-Problems hohe Bedeutung zu.

Kriterien für die Auswahl sind zeitliche und personelle Engpässe, häufiges oder regelmäßiges Auftreten, quantitativ oder qualitativ bessere Entscheidungsbasis, d.h. mehr relevante oder genauere Informationen und damit unmittelbar spürbarer Nutzen in konkreten Entscheidungssituationen. Dabei wird die Priorität am besten zunächst auf die Verkürzung der Aufbereitung und Bereitstellung prinzipiell im Unternehmen verfügbarer Informationen liegen, d.h. lediglich der zeitlichen Beschleunigung. Erst danach folgen Akquisition zusätzlicher, z.B. externer Informationen.

Die Reihenfolge sollte sich idealerweise an den wachsenden, durch zuvor bereitgestellte Informationen induzierten Nachforderungen orientieren. Um der Gefahr der Betriebsblindheit entgegen zu wirken, muß bei Operating Sponsor und tiefer in der Hierarchie Beteiligten ein Klima des aktiven Vorschlagens weiterer Informationen, Aufbereitungen bzw. Zusammenstellungen geschaffen werden. Außerdem ist dadurch eine Akzeptanzförderung seitens der in ihren ursprünglichen Machtpositionen beschnittenen Mitarbeiter zu erwarten.

3.5.2.4. EIS-Team

Aus dem zuvor Gesagten folgt unmittelbar eine interdisziplinäre Zusammensetzung des EIS-Entwicklungsteams, vertikal durch die gesamte Unternehmenshierarchie. Die Leitung muß dabei in der Fachabteilung, d.h. bei Executive und Operating Sponsor liegen. Zur automatischen, regelmäßigen Anbindung vorhandener, interner Datenbestände sind EDV-Spezialisten hinzu zu ziehen. Daneben muß mindestens eine weitere Spezialaufgabe im EIS-Team vertreten sein:

Im Hinblick auf die Verbreitung des EIS-Gedankens im Unternehmen ist frühzeitig eine organisatorische Instanz zur Standardisierung von EIS-Applikationen zu schaffen. Das betrifft einerseits die Auswahl der eingesetzten Softwarewerkzeuge zur Entwicklung der EIS-Anwendungen, andererseits aber auch die Festlegung von Gestaltungsrichtlinien der EIS-Oberflächen selbst. Dadurch können sowohl Wildwuchs der Darstellung gleicher Informationsinhalte verhindert und eine bessere Kommunikation zwischen EIS-Nutzern erzielt, als auch der Anpassungs- und Wartungsaufwand reduziert werden.

3.5.2.5. Infrastruktur

Aufbau und Betrieb von EIS erfordern eine adäquate Infrastruktur in personeller und sachlicher Hinsicht. Spezialaufgaben führen zur Einrichtung neuer Stellen oder aufgabenmäßiger Neuorientierung vorhanderer Stellen:

Zunächst muß eine neue Ebene der Informationsspeicherung, eine EIS-Datenbasis, geschaffen werden. Diese Funktion kann von einem Informationsmanager übernommen werden, dem Aufbau und Wartung des Unternehmens-Datenmodells obliegen. Schnittstellen in Form regelmäßiger Updates zur operativen Datenbasis müssen in Zusammenarbeit mit Datenbankadministrator und EDV-Abteilung organisiert und abgewickelt werden.

Vorauswahl, Bewertung und Einsatzunterstützung relevanter Software können von einem Benutzerservice-Zentrum, Information Center o.ä. übernommen werden. Bei Entwicklung und Anpassung sind zur Gewährleistung software-ergonomischer Prinzipien EDV-Spezialisten zumindest hinzuzuziehen.

3.5.2.6. Organisationsdurchdringung

In der ursprünglichen Zielsetzung konzentrierten und beschränkten sich EIS-Anwendungen ausschließlich auf die oberste Unternehmensebene. Informationen sollten direkt und an allen

als verzögernd oder gar filternd eingeschätzten hierarchischen Zwischenstufen vorbei, zur Führungskraft geleitet werden. Diese sollte gewissermaßen unabhängig gemacht werden.

Wenngleich dieses Argument zweifelsohne seine Berechtigung hat, so darf jedoch nicht übersehen werden, daß solchermaßen beschleunigter Informationsfluß nur dann seinen Nutzen entfalten kann, wenn die dadurch gewonnenen Erkenntnisse auch handlungsmäßig umgesetzt werden können. Und dies bedeutet Kommunikation mit eben den untergeordneten Abteilungen und Stellen, an denen die Information vorbei geleitet wurde. Kommunikation im Sinne von Akzeptanz übertragener Handlungsanweisungen setzt allerdings zumindest gleichen Informationsstand, und mehr noch Konsens über die Richtigkeit der den Anweisungen zugrundeliegenden Informationen voraus. Daraus folgt umittelbar, daß den Kommunikationspartnern die gleichen Informationen zur gleichen Zeit zur Verfügung stehen müssen. In der Regel wird dies nur einen Ausschnitt des Informationsvolumens der Führungskraft betreffen, dafür jedoch zusätzliche Möglichkeiten für Detailuntersuchungen erfordern. Im Idealfall wird untergeordneten Stellen im Rahmen der EIS-Hierarchie die Möglichkeit geboten werden können, einzelne Informationsaspekte mit verbalen Einschätzungen oder Vorkommentierungen zu versehen, ohne allerdings etwas an den reinen Daten verändern bzw. "manipulieren" zu dürfen.

Insofern folgt unmittelbar, daß erfolgreiche EIS tendenziell die gesamte Unternehmenshierarchie durchdringen müssen.

3.5.3. Technische Erfolgsfaktoren

3.5.3.1. Geschwindigkeit

Von den technischen Erfolgsfaktoren steht die Geschwindigkeit der EIS-Anwendung, d.h. das Antwortzeitverhalten an erster Stelle. Mit max. 1-2 Sekunden sind die Anforderungen dabei sehr hoch zu stecken. Angesichts der hohen Anforderungen durch die simultane Darstellung von Daten, Text, Bild und Graphik einerseits und die zumeist komplexen Abfragen mit umfangreichen Selektions- und Verdichtungsoperationen ergibt sich trotz immens gestiegener Verarbeitungskapazitäten derzeit verfügbarer Hard- und Software die Notwendigkeit zu architekturmäßigen Maßnahmen:

- ❒ *Zwischenspeicherung ausgewählter Informationen*
- ❒ *Vorverdichtung*
- ❒ *Speicherung verdichteter Informationen*
- ❒ *partielles Downloading*

3.5.3.2. Flexibilität

Neben einem relativ kleinen Anteil fixer Informationen zeichnet sich der Informationsbedarf von Führungskräften durch eine hohe Dynamik aus. Der Erfolg eines EIS hängt entscheidend davon ab, daß neue Informationsbedürfnisse schnell integriert werden können. Dies erfordert einerseits, daß ein breites Spektrum von Informationsquellen mit leistungsfähigen Selektionsmechanismen angeschlossen werden kann, andererseits benötigte Selektionen leicht und schnell durchgeführt und in eine präsentationsreife Darstellung und Bedienungsoberfläche umgesetzt werden können. Das impliziert unmittelbar, daß Anpassung und Weiterentwicklung zu einem großen Teil in den Händen von Fachspezialisten, möglichst der Operating Sponsors, d.h. der persönlichen Assistenten und Referenten der Führungskräfte liegen müssen.

3.5.3.3. Entwickler-Produktivität

Leichte Bedienbarkeit der Entwicklungsoberfläche durch Fachspezialisten kann auf lange Frist betrachtet nur den geringsten Beitrag für eine hohe Produktivität bei der Bewältigung der Flexibilitätsanforderungen von EIS liefern. Viele EIS-Anwendungen weisen eine hohe strukturelle Ähnlichkeit auf und unterscheiden sich lediglich durch die Menge der präsentierten Daten: Als Beispiele sind Zeitreihenbetrachtungen für unterschiedliche Regionen, Produkte oder Aggregationen darauf für verschiedenste Kennzahlen zu nennen, von Absatzzahlen über Umsätze und Deckungsbeiträge bis hin zu Planabweichungen usw.. Die Anzahl generierter Präsentationsbildschirme erreicht in der Praxis schnell die 1000er-Marke. Kleinste Anpassungen in jedem dieser strukturell gleichen Schirme, z.B. Änderungen des Graphiktyps oder Ergänzung um eine prozentuale Abweichung bis hin zur Erweiterung aller Schirme um den Abruf einer weiteren Funktion oder Sprungmöglichkeit dürfen nicht dazu führen, alle 1000 oder mehr Bildschirmspezifikationen einzeln ändern zu müssen. Datengetriebenheit ist hier das Stichwort.

Ein einheitlicher, unternehmensweiter Bildschirmaufbau ist nicht nur aus den o.g. Kommunikationsgründen sondern auch im Interesse der Gewährleistung software-ergonomischer Prinzipien notwendig. Bei zunehmend angestrebtem Einsatz von Fachspezialisten als EIS-Entwickler kann jedoch eine diesbezügliche Vorbildung nicht vorausgesetzt werden. Eine entsprechende Unterstützung durch die Entwicklungssoftware kann die Entwicklungs-Qualität erhöhen helfen: Im einfachsten Fall werden alternative Gestaltungsmuster zur Auswahl zur Verfügung gestellt, in einer weiteren Ausbaustufe wird einem herausgehobenen Entwicklerkreis die Möglichkeit geboten, selbst solche Muster für das Unternehmen zu definieren. Im Idealfall sind Ausbaustufen denkbar, in denen dynamische, kontext-abhängige Re-

striktionen für den Bildschirmaufbau, die Farbwahl usw. den Entwicklungsprozess überwachen und bei Bedarf erklärend und empfehlend einschreiten.

3.5.3.4. Anwender-Akzeptanz

Neben Geschwindigkeit in Betrieb und Anpassung stehen für die Akzeptanz durch den Anwender primär zunächst gestalterische Aspekte der Informationspräsentation, -abfrage und -weiterverarbeitung im Vordergrund:

- ❒ Leichte, d.h. intuitiv verständliche Bedienbarkeit und damit schnellste (Wieder-) Erlernbarkeit infolge der unregelmäßigen und zumeist zeitlich nur kurzen Verwendung sind durch wenige, optische und eindeutige Symbole auf der Bildschirmoberfläche sicherzustellen.
- ❒ Interpretationssicherheit, Relevanz und Aussagekraft dargebotener Informationen müssen dadurch gewährleistet werden, daß der Informationsgehalt der elektronischen Präsentation der herkömmlichen, zumeist mündlichen oder schriftlichen über persönliche Berichterstatter möglichst wenig nachsteht. Dies bedeutet, daß erklärende und bewertende Zusatzinformationen problemlos und individuell nachgefordert werden können. Das Spektrum dieser Informationsfacetten ist in Abb. 3-4 am Beispiel einer typischen, aggregierten Kennzahl dargestellt:

  "durchschnittliche Umsatzänderung im 3. Quartal 1993= +4%"

Die potentiell benötigten Zusatzinformationen zu einer korrekten Interpretation dieser Kennzahl können wiefolgt klassifiziert werden:

- ❒ *Definition*

  Zuallererst muß Klarheit bezüglich der definitorischen Zusammenhänge bestehen. Welcher Umsatz ist gemeint, z.B. Brutto- oder Netto-, bzw. auf Basis welcher Bewertungsgröße, d.h. worauf bezieht sich die Durchschnittsbildung, z.B. Produkte, Regionen oder die Zeitquartale, was ist Bezugsgröße für die Änderung, das 3. Quartal des Vorjahres oder das 2. Quartal des aktuellen Jahres oder vielleicht die Plangröße für das 3. Quartal des aktuellen Jahres?

  Sicherlich bilden sich in jedem Unternehmen hierfür Konventionen des eigenen Sprachgebrauchs heraus, dennoch bergen solche per Unternehmenskonvention gebildeten "Abkürzungen" immer wieder die Gefahr zu unterschiedlicher Interpretation und damit zu Kommunikationsproblemen in sich. Da von Seiten der Datenadministration im Zusammenhang mit Unternehmensdatenmodellen und Datenbanksystemen solche definitorischen Klarstellungen zunehmend bereits in Form von Data Dictionaries erfaßt und verwaltet werden, bietet sich das Angebot eines bedarfsweisen Rückgriffs aus dem EIS heraus an.

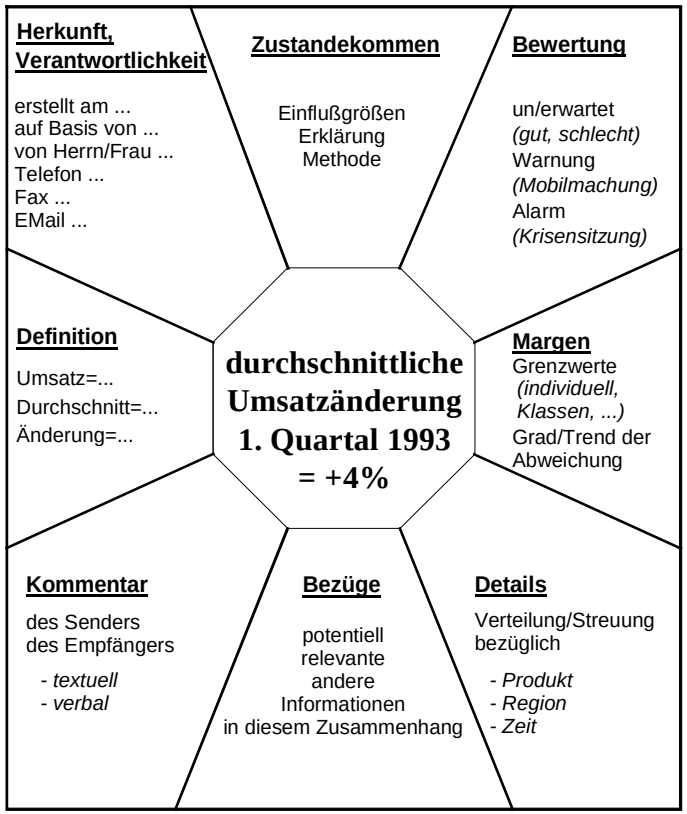


Abb. 3-4: Facetten notwendiger Zusatzinformationen

- ❒ *Herkunft, Verantwortlichkeit*

  Zur Einschätzung der Bedeutung respektive der Ableitung eventueller weiterer Aktionen, werden häufig dynamische Zusatzinformationen benötigt: hierunter fallen bspw. das Stichdatum oder eine Aussage darüber, auf welcher Basis die Kennzahl gewonnen wurde. Vor allem in multinationalen oder Holding-Gesellschaften laufen die Teilergeb-

nisse nicht gleichzeitig sondern über mehrere Tage oder gar Wochen verteilt ein; um nicht bis zum letzten, häufig relativ bedeutungslosen Wert warten zu müssen, werden vorzeitig ein oder mehrere Teilergebnisse produziert; Anreicherungen mit Zusatzinformationen über die verwendete Basis des Teilergebnisses, z.B. Anteil Werte oder Volumen in Bezug zu einer repräsentativen Vorperiode erlauben frühzeitigeres Reagieren. Für derartige und weitere Rückfragen ist es erforderlich, schnell Verbindungen zu verantwortlichen Mitarbeitern im Unternehmen herstellen zu können, notfalls auch während deren Abwesenheit: hierzu sind Informationsabhängig Namen, Abteilungen, Telefon- oder Telefaxnummern bzw. EMail-Adressen im Hintergrund bereitzuhalten.

❒ *Zustandekommen*

In engem Zusammenhang mit potentiellen Auskunftspersonen über im EIS präsentierte Daten sind Fragen nach den Ursachen für das Zustandekommen dieser Werte zu sehen. Bei definitorischen, rechnerischen Zusammenhängen etwa in Folge von Aggregation oder Kennzahlenbildung kann der EIS-Anwender sich diese "Nachfrage" teilweise automatisiert selbst beantworten: Mittels "Drill-Down"-Technik können schrittweise die jeweils unmittelbaren Einflußgrößen abrufbar bereitgestellt werden.

In einer weiteren Ausbaustufe läßt sich die Menge der in Frage kommenden Einflußgrößen mittels einfacher Analyseregeln automatisiert auf das Maß der relevanten Größen reduzieren; ein Beispiel hierzu stellt das EXPLAIN-Kommando der Planungssprache IFPS/Plus dar, das hierzu eine implizites regelbasiertes System integriert hat (vgl. Kap. 6.2.2.3 und [Rieger 90, Wissensbasierte Planungssprachen]).

Neben den unmittelbaren Einflußgrößen ist häufig auch die Art der Verknüpfung, d.h. bspw. die formelmäßige Kennzahlendefinition von Interesse. Darüber hinaus kommen verbale Kommentare durch Informationslieferanten in Frage, zu denen dann der Kontakt über die Zusatzinformationen aus der zuvor beschriebenen Gruppe herangezogen werden kann.

❒ *Bewertung*

Fragen des Zustandekommens vorgelagert ist die Bewertung präsentierter Informationen: Ist die Kennzahl erwartet oder unerwartet gut oder schlecht? Ist die Abweichung von den Erwartungen als Warnhinweis oder gar als Alarmzeichen zu werten? Eine optische Markierung von Informationen bezüglich dieser Eigenschaften kann dazu beitragen, das Augenmerk und damit das Entscheidungsverhalten von Führungskräften direkt und unmittelbar auf relevante und wesentliche Informationseinheiten zu lenken. In Verbindung mit einer automatischen Vorselektion solchermaßen gekennzeichneter Größen durch das EIS sind erhebliche Reduktionspotentiale der Informationsflut zu erreichen. Auf die verschiedenen Formen dieses Exception-Reporting wird in Kap. 5.3.5 detailliert eingegangen.

❒ *Margen*

Voraussetzung für eine funktionsfähige, automatisierte Bewertung ist die Möglichkeit, Margen für die einzelnen Bewertungsausprägungen spezifizieren zu können. Im einfachsten Fall sind dies numerische Intervalle für einzelne Kennzahlen. Um auch komplexere Situationseinschätzungen in die Bewertung einbeziehen zu können, ist es notwendig, mehrere solcher Einzelintervalle logisch miteinander verknüpfen zu können. In einer weiteren Ausbaustufe sind auch qualitative und unscharfe Margen denkbar, was in den Bereich des Einsatzes von Techniken wissensbasierter Systeme verweist.

Neben der Feststellung einer Grenzwertverletzung spielen für die Beurteilung dieser Abweichung auch Grad und Häufigkeit im Zeitverlauf eine große Rolle, also die Frage, ob es sich um einen einmaligen, vielleicht nur kleinen Ausreißer handelt oder ob dieser einem längerfristigen Trend folgt. In diesem Falle wäre ein frühzeitiger Hinweis, bereits vor Erreichen oder Überschreiten der gesetzten Margen, von Nutzen. Auf diese Aspekte wird ebenfalls in Kap. 5.3.5 im Zusammenhang mit Konstrukten von EIS-Generatoren bezüglich des Exception Reporting näher eingegangen.

❒ *Details*

Zur Analyse des Zustandekommens aggregierter Informationseinheiten wie auch auffallender Grenzwertabweichungen ist es erforderlich, Detailinformationen bedarfsweise leicht abrufen zu können. Insbesondere Verteilung und Streuung der Abweichung stehen im Vordergrund des Interesses: Handelt es sich bspw. um eine homogene, d.h. gleichgerichtete und größenmäßig gleiche Abweichung über alle Produkte bzw. Regionen? Gibt es einzelne oder einige wenige, hauptverantwortliche Ausreißer? Oder heben sich gar positive und negative Abweichungen einzelner Produkte und Regionen gegenseitig auf?

Insbesondere in letzterem Fall muß sichergestellt werden, daß diese Unregelmäßigkeiten auf Detailebene nicht in dem primär auf die Sichtung aggregierter Daten ausgerichteten EIS verlorengehen. Es müssen im Rahmen des Exception Reporting also Mechanismen vorgesehen werden, die auch sich kompensierende Detailabweichungen quasi nach oben durchreichen. Mögliche generische Lösungskonstrukte zu dieser Problematik werden unter den Schlagworten Drill-Down und Exception-Reporting in den Kapiteln 5.3.4 und 5.3.5 detailliert behandelt.

❒ *Bezüge*

Bisher wurden im Umfeld der Abfrage von Zusatzinformationen ausschließlich "harte" definitorische Zusammenhänge angesprochen. Wesentlich für Entscheidungen von Füh-

rungskräften ist jedoch das Zusammenführen von für eine konkrete Entscheidungssituation relevanten Informationen unterschiedlichster Art, z.B. das Hinzuziehen von Branchenvergleichsdaten oder externen Informationen, z.B. über die allgemeine Wirtschaftslage, Aktienkurse etc..

Diese Pakete jeweils potentiell relevanter Informationen können entscheidungstypabhängig, zentral oder individuell zusammengestellt und über eine Typisierung der Präsentationsbildschirme diesen zugeordnet und damit der Führungskraft zugänglich gemacht werden. Als Technik kommt insbesondere der Hypertext-Ansatz in Betracht.

❒ *Kommentare*

Als Alternative zum Angebot individueller Nachfragen durch die Führungskraft über abrufbare Hinweise auf Verantwortlichkeiten kommen bereits mitgelieferte Kommentare in textueller oder verbaler Form in Frage. Hier lassen sich im Vorfeld bereits Erläuterungen, z.B. erklärender Art über eingetretene Abweichungen und deren Ursachen hinterlegen. Außerdem bietet dieses Instrument eine Möglichkeit der Integration untergeordneter Stellen, einen unter Gesamtakzeptanzgründen hoch einzuschätzenden Faktor.

Auf dem gleichen Wege lassen sich so auch Nachfragen, zusätzliche Analyseaufträge oder Erklärungswünsche bis hin zu konkreten Handlungsanweisungen in Form von textuellen Kommentaren des Empfängers einbauen. Schließlich ist diese individuelle Kommentierungsmöglichkeit für den Empfänger von hohem Interesse, z.B. als Merkposten für geplante, regelmäßige oder adhoc einzuberufende Besprechungen. In diesem Zusammenhang tritt auch der Bedarf zu individuellen Markierungen und Zusammenstellungen ausgewählter EIS-Informationseinheiten als Vorbereitung solcher Besprechungen auf. In Verbindung mit einer Wiedervorlagefunktion, einer speziellen Ausprägung des EMail, findet sich dies in EIS-Werkzeugen häufig unter dem Schlagwort der Kalenderfunktion wieder. Hierauf wird detailliert in Kap. 5.5.2 eingegangen.

3.5.3.5. Anwendungs-Individualität

Weitere wesentliche Erfolgsfaktoren von EIS sind alle technischen Gestaltungsmaßnahmen, die zur Individualisierung der Anwendung beitragen. Hierunter sind sowohl persönliche Bedienungsalternativen, z.B. die Steuerung wahlweise per Tastatur, Maus oder Touchscreen zu verstehen als auch vor allem funktionale Möglichkeiten. Diese reichen von der Anpassung der Präsentationsoberfläche über die Zusammenstellung und Anordnung der dargestellten Informationseinheiten bis hin zur Bildung neuer, persönlicher Gruppierungen von Informationseinheiten, z.B. zum Zwecke von Besprechungen, Berichten oder Handlungsanweisungen.

Aus Gründen der oben geforderten Entwicklungsproduktivität darf Individualität nicht dazu führen, daß für jeden Anwender vollständig eigenständige Oberflächen erzeugt werden. Vielmehr sind für Typen von Informationseinheiten jeweils (wenige) alternative Präsentationsformen zu suchen und zu realisieren, die dann per Zuordnung zu ebenfalls (wenigen) Anwendertypen individuell aktiviert bzw. auch ausgetauscht werden können.

3.5.3.6. Sicherheit

Führungsinformationssysteme enthalten per definitione in komprimierter Form höchst aussagekräftige Informationen über den internen Zustand eines Unternehmens sowie dessen aktuelle Stärken und Schwächen im Vergleich zur Konkurrenz. Eine Chance auf Akzeptanz, d.h. tagtäglichen Einsatz eines solchen Systems sowie die Bereitschaft, diese Informationen über das elektronische Medium EIS auszutauschen, kann nur erreicht werden, wenn umfangreiche Schutz- und Sicherungsbedürfnisse zuverlässig erfüllt werden (vgl. [Bullinger 93, Führungsinformationssysteme], S. 57).

Das Spektrum notwendiger Sicherungsmaßnahmen umfaßt die persönliche Zugriffssteuerung je Datenelement über alle Verdichtungs- und Gruppierungsstufen in Form von EIS-Bildschirmen ebenso wie funktionale Weiterverarbeitungen auf elektronischem Wege. Aus Gründen der Wartungsproduktivität ist nach Wegen zu suchen, die systemimmanent automatisch, z.B. in Form zwangsweise auszufüllender Attribute der verschiedenen Informationseinheiten, eine Spezifikation von Zugriffsattributen bzw. Auskünfte über alle Berechtigte je Informationseinheit gewährleisten.

4. Das Konzept des rechnerunterstützten Arbeitsplatzes (RAP)

4.1. Motivation des Konzepts

Das in Kapitel 3 beschriebene Konzept der Executive Information Systems (EIS) stellt zunächst nur eine funktionale Leistungsspezifikation für die Rechnerunterstützung eines bestimmten Arbeitsplatztyps in Unternehmen, den der Führungskräfte, dar. Sie beinhaltet noch keine Festlegungen bezüglich der DV-technischen Umsetzung. Angesichts der Dynamik der Anforderungen scheidet der Einsatz von Programmiersprachen der 3. und Softwareentwicklungsumgebungen und 4. Generation allerdings praktisch aus.

Zur Erfüllung der kritischen Erfolgsfaktoren Entwicklungsproduktivität und Flexibilität kommt für die Realisierung von rechnergestützten Arbeitsplätzen für Führungskräfte vielmehr praktisch ausschließlich der Einsatz spezialisierter Entwicklungswerkzeuge in Betracht. Diese zeichnen sich dadurch aus, daß typische Strukturen und Funktionen einer Klasse von Aufgaben vorprogrammiert sind und als parametrisierte *Konstrukte* für die Implementierung konkreter Aufgaben zur Verfügung gestellt werden. Der Schwerpunkt der Implementierung verlagert sich damit von der *Programmierung* hin zur *Spezifikation.* Anwendungen werden lediglich durch Kombination dieser generischen Konstrukte und durch Spezifikation der vorbereiteten Parameter konfiguriert.

Je nach Ausmaß der Parameter wird hierdurch zwar die erreichbare Anwendungsindividualität eingeschränkt, die Produktivitätsvorteile insbesondere in Bezug auf Wartung und Weiterentwicklung überwiegen diesen Nachteil jedoch bei weitem. Aufgrund der leichteren Bedienbarkeit solcher Spezifikationswerkzeuge kann die Anwendungsentwicklung zunehmend von der EDV-Abteilung weg und hin zur Fachabteilung verlagert werden. Im Extremfall bedeutet dies, daß nicht eine Implementierung auf Basis eines Entwicklungswerkzeugs sondern das Entwicklungswerkzeug selbst zum eigentlichen Hilfsmittel am Arbeitsplatz wird.

Das Angebot in Frage kommender Entwicklungswerkzeuge hat sich historisch zunächst eher funktional und aufgabenorientiert denn arbeitsplatzorientiert entwickelt, d.h. nicht das jeweils gesamte Aufgabenspektrum von Sachbearbeitern, Sekretariaten, Controllern oder Führungskräften stand im Vordergrund, sondern allgemeine Funktionen wie Datenbankabfrage, Graphik- und Berichtsgenerierung, Electronic Mail, Kontenführung, Textverarbeitung usw.. Wie die Analyse in Kapitel 5 zeigen wird, gilt dies auch für die seit Mitte der Achtzi-

ger Jahre entwickelten Werkzeuge für Executive Information Systems (EIS-Generatoren), insbesondere wenn man das gesamte Anforderungsprofil aus Kapitel 3 berücksichtigt. Andererseits werden zahlreiche EIS-Funktionen bereits mehr als ausreichend durch das Leistungsspektrum vorhandener Entwicklungswerkzeuge abgedeckt. Es kann auch festgestellt werden, daß EIS-Generatoren zunehmend durch entsprechende Schnittstellen genau davon Gebrauch machen.

So können *Datenbanksysteme* die Organisation, Speicherung und Verwaltung großer Datenmengen leisten. *Abfragesprachen* können deren Selektion, Aggregation und Verknüpfung übernehmen, *Reportsysteme* tabellarische und graphische Formatierungsleistungen für die Ergebnispräsentation liefern. *Planungssprachen* sind geeignet, modellmäßige Verknüpfungen zwischen Daten mit Hilfe von Definitions- und Verhaltensgleichungen zu realisieren sowie vergleichende Alternativ- oder optimierende Zielrechnungen zu produzieren. Schließlich können *wissensbasierte Systeme* klassifikatorische, konfigurierende oder interpretierende Fragestellungen auf Basis heuristischer Zusammenhänge beantworten helfen.

Zur gesamtheitlichen Rechnerunterstützung von Arbeitsplätzen jeglicher Art, also nicht nur für Führungskräfte, ergeben sich somit 2 Alternativen:

- ❒ Die Kombination aus Teilanwendungen, die jeweils mit größtenteils am Markt vorhandenen, hochspezialisierten Entwicklungswerkzeugen realisiert sind, oder
- ❒ die Integration aller benötigten Funktionalitäten in jeweils ein Entwicklungswerkzeug je Arbeitsplatztyp oder Typ von Aufgabenkonglomerat.

Es ist jedoch festzustellen, daß sich im zweiten Fall der Spezialisierungsgrad der Einzelkomponenten und damit auch deren Produktivität häufig erheblich verringert; zumindest jedoch wird diese Entwicklungsrichtung mit zunehmender Zahl integrierter Funktionen tendenziell gefördert. Deshalb soll diese Art Softwarepakete, die sich auf dem (bereits mehrfach gescheiterten) Pfad zum general-problem-solver zu bewegen scheint, hier nicht weiter betrachtet werden.

Vielmehr wird nach Wegen gesucht, wie hochspezialisierte und leistungsfähige Werkzeuge logisch und entwicklungsmäßig integriert werden können, ohne daß der bei den Einzelkomponenten bewährte Produktivitätsvorteil vorgefertigter, spezialisierter, generischer Anwendungskonstrukte verloren geht. Dies bietet ferner den Vorteil, beliebige Arbeitsplatztypen auch mit unterschiedlichen funktionalen Schwerpunkten in einem Konzept unterstützen zu können, indem bei Bedarf einfach zusätzliche, am Markt zumeist und zunehmend vorhandene Werkzeuge hinzugefügt oder ausgespart werden. Gerade dieser zweite Aspekt trägt dazu bei, den häufig unnötigen, belastenden Overhead integrierter Entwicklungswerkzeuge zu vermeiden.

Ein derartiges generelles Komponentenkonzept des "rechnerunterstützten Arbeitsplatzes (RAP)" auf der Basis der Integration von Entwicklungswerkzeugen erfordert zum einen ein geeignetes Klassifizierungskriterium zur Identifikation generischer Werkzeuge und zum anderen eine detaillierte Beschreibung aller Integrationsebenen, um entsprechende Schnittstellendefinitionen für die Werkzeuge spezifizieren zu können.

4.2. Die Komponentenklassen

4.2.1. Wahl des Klassifizierungskriteriums

Die Wahl von Klassifizierungskriterien muß sich grundsätzlich am verfolgten Ziel orientieren. Im vorliegenden Fall steht die Integration von Teilanwendungen, die auf Basis von Werkzeugklassen implementiert sind, im Zentrum des Interesses. Dabei soll der Integrationsprozess selbst genauso unterstützt werden, wie die Implementierung der Teilanwendungen, d.h. auf der Ebene von Spezifikationen unter weitestgehender Verwendung generischer Konstrukte. Daraus folgt, daß die gesuchten Klassifizierungskriterien geeignet sein müssen, Anwendungen im Sinne von Teilaufgaben zu bilden.

Somit scheiden zumindest auf dieser obersten Klassifizierungsebene gebräuchliche Kriterien methodischer Art, wie z.B. konventionell vs. wissensbasiert oder Datenbank- vs. Spreadsheet-orientiert aus. Das gleiche gilt aus einem anderen Grund aber auch für die vielleicht naheliegende Unterteilung nach betrieblichen Funktions- und Anwendungsbereichen. Danach könnte bspw. unterteilt werden in Auftragsstatistik, Lagerumschlagsstatistik, kurzfristige Erfolgsrechnung oder ROI-Analyse. Dieses Kriterium erscheint ungeeignet, da Gemeinsamkeiten im Sinne generischer Problemlösungskonstrukte nicht berücksichtigt werden. Bspw. verwenden alle genannten Aufgaben aggregierende Modelle auf Basis von Definitionsgleichungen.

Jedem Arbeitsschritt bzw. jeder Teilaufgabe, egal in welchem funktionalen Unternehmensbereich, können aber eine oder mehrere Phasen des Problemlösungsprozesses zugeordnet werden. Somit lassen sich alle zu unterstützenden Aufgaben in ein relativ einfaches, allgemeingültiges Phasenschema einordnen.

Durch schrittweise Verfeinerung der Phasen des Problemlösungsprozesses können prinzipiell beliebig detaillierte Klassenhierarchien gebildet werden. Umgekehrt lassen sich Pro-

blemlösungsphasen aber auch zu Oberklassen, sogenannten Problemtypen aggregieren. Auf diese Weise gelangt man zu den hier verwendeten Komponentenklassen des *Decision Support* und *Data Support.*

4.2.2. Data Support

Unter *Data Support* sollen alle Unterstützungswerkzeuge für frühe Phasen von Problemlösungsprozessen verstanden werden. Darunter fallen insbesondere Problemerkennung, -identifikation und -analyse. Ergebnis des Data Support ist generell die Feststellung eines Handlungs- bzw. Entscheidungsbedarfs.

Bspw. ergibt sich aus dem beobachteten Anstieg von Lieferzeiten die Notwendigkeit zur Ursachenanalyse. Die anschließende Auswertung potentieller Einflußgrößen wie Auftragseingängen, Lagerbeständen oder Durchlaufzeiten kann Handlungs- und damit Entscheidungsbedarf bezüglich der Bestellpolitik identifizieren.

Typische Kennzeichen für Data Support sind somit Überwachung, Kontrolle und Analyse von internen und externen Daten. Somit stehen Datenretrieval und anschließende Aggregationsfunktionen im Vordergrund.

4.2.3. Decision Support

Unter *Decision Support* hingegen sollen alle Unterstützungswerkzeuge für spätere Phasen von Problemlösungsprozessen verstanden werden. Dazu gehören insbesondere Formulierung von Entscheidungsmodellen, Alternativengenerierung, -bewertung und -auswahl. Ergebnis des Decision Support sind somit bewertete Vorschläge für unternehmerische Handlungen.

Bspw. könnte Decision Support in obigem Beispiel in Form von Aussagen über die Konsequenzen alternativer Parameter der aktuellen Bestellpolitik, aber auch alternativer Bestellpolitiken überhaupt bestehen. Die Auswirkungen dürften sich dabei nicht nur auf die Korrektur der beobachteten Lieferzeitproblematik beschränken, sondern müßten auch kostenmäßige Konsequenzen sowie Seiteneffekte auf andere Unternehmensbereiche, wie z.B. Lagerkapazitäten, Versand, Personal und Organisation berücksichtigen.

Typische Kennzeichen für Decision Support sind somit Planung und Steuerung. In jedem Fall erfordert dies als Basis ein umfangreiches Modell der relevanten Unternehmensteile und -zusammenhänge.

4.3. Die Integrationsebenen

4.3.1. Phasen-Integration

Durch die Zerlegung der Implementierung von rechnergestützten Hilfsmitteln für das Aufgabenspektrum einzelner Arbeitsplätze im Interesse der weitestgehenden Nutzung spezialisierter Entwicklungswerkzeuge entsteht umgekehrt Integrationsbedarf auf verschiedenen Ebenen. Der offensichtlichste und einfachste wird durch die Trennung in Data und Decision Support ausgelöst. Voraussetzung für diesen Integrationsbedarf ist jedoch eine entsprechende Stellenbildung im Sinne der Bearbeitung ganzer Vorgangsketten an jeweils einem Arbeitsplatz, d.h. Überwachung, Analyse und Steuerung im Sinne der Parametereinstellung der Bestellpolitik aus obigem Beispiel müssen in einer Hand liegen.

In diesem Falle sind beide Anwendungen und damit auch die dazu gehörigen (Entwicklungs)-Werkzeuge am Arbeitsplatz zu integrieren. Während im Falle der Verteilung auf verschiedene Arbeitsplätze lediglich der Zugriff auf gemeinsame Daten gewährleistet sein muß, kommt hier die Forderung nach einer homogenen, zumindest ähnlichen, d.h. gemeinsamen Grundprinzipien gehorchenden Bedienungsoberfläche hinzu.

Diese Form der Integration ist als ausschließlich horizontal, d.h. sequentiell entlang der Phasen des Problemlösungsprozesses verlaufend, auf gleicher organisatorischer Ebene liegend zu charakterisieren.

4.3.2. Organisatorische Integration

Häufig orientiert sich jedoch die Aufgabenverteilung und Stellenbildung im Unternehmen nicht an dem Prinzip der Vorgangsketten, sondern am Prinzip der Verantwortung, d.h. dem Gewicht der mit den Entscheidungen verbundenen Bedeutung für das Unternehmen.

Im Beispiel der Bestellpolitik beschränkt sich etwa der Entscheidungsspielraum auf der untersten, operativen Ebene darauf, den aktuellen Lagerbestand im Bezug zu vorgegebenen Schwellwerten zu beobachten und bei Unterschreitung entsprechend ebenfalls vorgegebenen Entscheidungsparametern, z.B. einer Mindestbestellmenge, Lieferantenprioritätenliste usw. aufzustocken. Initiativen zur Änderung der Parameter dieses impliziten Entscheidungsmodells können parallel durch Beobachtung aggregierter Daten auf nächst höherer, dispositiver Organisationsebene ausgelöst werden. Zumeist geschieht dies jedoch in Form einer administrativ organisierten Berichtsaufgabe der operativen an die dispositive Ebene, so daß sich der Data Support der dispositiven Ebene auf die Selektion von Ausnahmen sowie deren Analyse

und Bewertung beschränken kann. Zur Unterstützung dieser Schritte können bei dieser Aufgabenteilung zwischen den organisatorischen Ebenen insbesondere Zusatzinformationen eingebaut werden, z.B. erläuternde Kommentare der operativen Ebene.

Neben der horizontalen Integration zwischen den Klassen Data und Decision Support tritt hier also eine vertikale Integration zwischen Arbeitsplätzen auf unterschiedlichen Ebenen der Unternehmenshierarchie hinzu.

Wird die Berichtsfunktion z.B. zum Zwecke der Zusammenfassung verschiedener Berichtsteile stellenmäßig zusammengefaßt, etwa in Form einer Controlling-Abteilung, so erweitert sich der Aspekt der horizontalen Integration in zwei Unterfälle, zum einen die bekannte Zusammenführung innerhalb eines Arbeitsplatzes, zum anderen die Zusammenführung zwischen Arbeitsplätzen auf gleicher organisatorischer Ebene.

Die verschiedenen Formen der organisatorischen Integration sind in Abb. 4-1 zusammenfassend dargestellt. Zusätzlich zu den rechnergestützten Hilfsmitteln, die jeweils als Rechtecke dargestellt und bezüglich ihrer Problemlösungsklasse in Data Support (EIS) und Decision Support (DSS) unterteilt sind, enthält die Abbildung auch vor-, zwischen- und nachgeschaltete nicht rechnergestützte Arbeitsgänge. Diese sind durch Kreise repräsentiert. Jeder gerichtete Pfeil stellt eine Kommunikationsbeziehung in Form eines Informationsaustausches dar und steht im Falle der Verbindung von zwei rechnergestützten Komponenten für einen organisatorischen Integrationsbedarf.

4.3.3. Methoden-Integration

Die zuvor beschriebenen Integrationsebenen werden in der Literatur zumeist nicht explizit sondern nur implizit im Zusammenhang mit der Integration von Teilanwendungen diskutiert, die auf methodisch unterschiedlichen Basen realisiert sind. Ein Beispiel ist die Verknüpfung von Unternehmensmodellen mit Datenbanken, d.h. die Auslagerung der Modelldaten in ein Datenbanksystem. Ein anderes häufig anzutreffendes Beispiel ist die Ergänzung von konventionellen Systemen, z.B. Planungsmodellen, mit wissensbasierten Komponenten, z.B. zur Analyse von hauptverantwortlichen Einflußgrößen (vgl. [Rieger 90, Wissensbasierte Planungssprachen]).

Die Bedeutung dieser Integrationsebene ergibt sich primär dadurch, daß Entwicklungswerkzeuge aufgrund der starken Differenzen der verwendeten Konstrukte zumeist methodisch orientiert sind. Trotz der unbestritten bestehenden technischen Problematik wird diesem Aspekt aber im folgenden nur eine untergeordnete Rolle beigemessen, da sich methodische

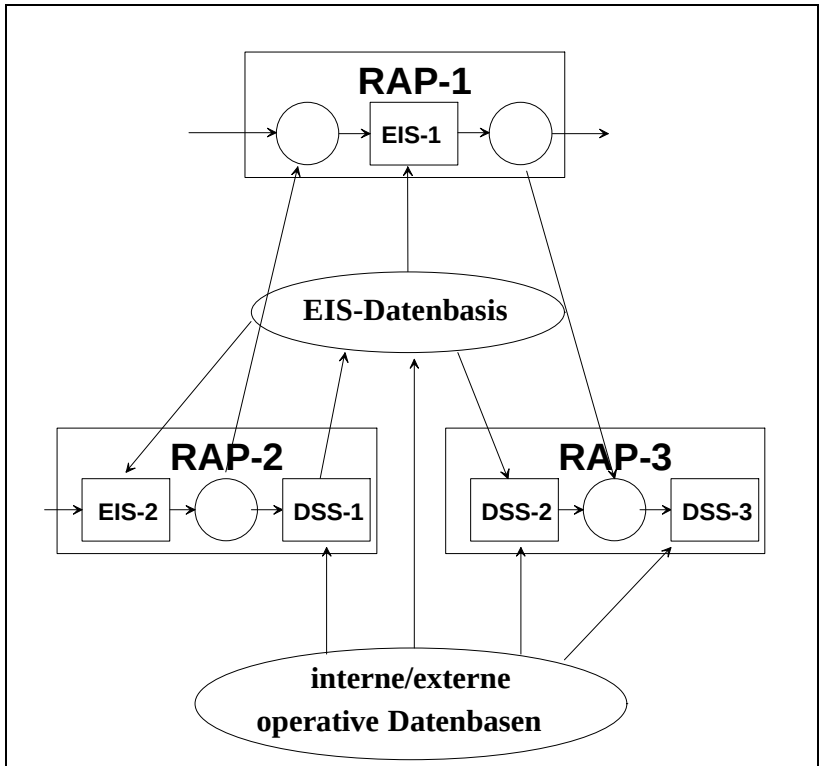


Abb. 4-1: organisatorische Integration

Integration immer auch auf die beiden anderen Integrationsformen zurückführen läßt, also als ein Teilproblem der Phasen- bzw. organisatorischen Integration darstellen läßt, was umgekehrt nicht gilt.

So kann bspw. eine wissensbasierte Lieferantenauswahl im genannten Beispiel aus dem operativen Bestellwesen als Teilaufgabe in einem horizontalen Integrationsprozess interpretiert werden. Umgekehrt kann der wissensbasierten Methode keine übergreifende Rolle bei der Rechnerunterstützung des Arbeitsplatzes zugesprochen werden, im Gegenteil konkurriert sie mit alternativen konventionellen Methoden in der gleichen Phase des Problemlösungsprozesses.

Diese Sichtweise ist in Abb. 4-2 zusammenfassend illustriert. In der Vertikalen sind von unten nach oben die Phasen des Problemlösungsprozesses mit typischen Bezeichnungen der Aufgabentypen aus dem konventionellen (links) resp. wissensbasierten (rechts) Methodenbereich aufgetragen. *Analyse* korrespondiert mit *Diagnose* bzw. *Klassifizierung*, *Optimierung* als Vertreter der Alternativengenerierung mit *Konfiguration* und *Simulation* als Vertreter der Alternativenbewertung und -auswahl mit *Interpretation.*

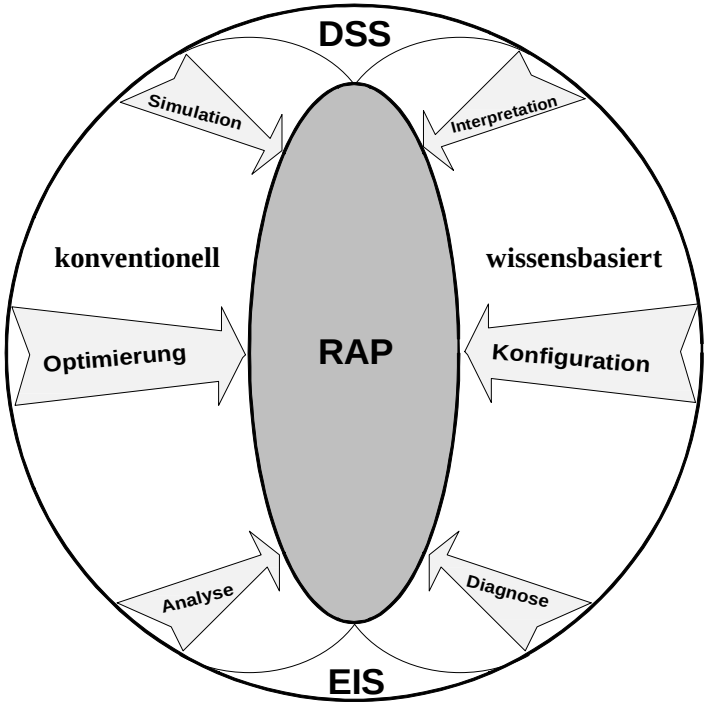


Abb. 4-2:	Phasen-/Methoden-Integration

Zusätzlich repräsentiert die obere Hälfte die Klasse der Decision Support Systeme, während die untere dem Data Support (EIS) zuzuordnen ist. Die Übergänge sind dabei fließend. Im Zentrum dieses Schaubilds steht naturgemäß der zu versorgende rechnerunterstützte Arbeitsplatz. Je nach hierarchischer Positionierung bzw. aufgabenmäßiger Zuordnung ist er nun aus Anwendungssystemen der verschiedenen Problemlösungs- bzw. Methodenklassen zusammenzusetzen. Im Falle von Führungskräften bzw. leitenden Angestellten kann der in der Literatur zu findende Begriff der Executive Support Systeme (ESS) problemlos und konsistent integriert werden. Mehr noch, ESS beschreiben aufgrund der bereits in der Anforderungsbeschreibung in Kapitel 3 ablesbaren Tendenz zur Integration auch entscheidungs-unterstützender Funktionen in EIS exakter das Ziel rechnerunterstützter Arbeitsplätze für Führungskräfte.

4.4. Entwicklungswerkzeuge

Ausgehend von diesem definitorischen Rahmenkonzept lassen sich nun die Werkzeugklassen EIS- sowie DSS-Generatoren identifizieren:

EIS-Generatoren reduzieren sich dabei durch die Einführung der Begriffe RAP und ESS - übrigens in Übereinstimmung zu dem tatsächlichen Leistungsumfang angebotener Werkzeuge am Markt - zu Entwicklungstools für das Spektrum des Data Support im Bereich von Führungskräften.

Dem gegenüber stehen Werkzeuge des Decision Support, die Funktionen der quantitativen Modellbildung und Kalkulation mit berichtsmäßiger und graphischer Ergebnisaufbereitung vereinen. Aufgrund der Integration analytischer Zusatzfunktionen zur Beantwortung von Fragestellungen wie What-if oder What-to-do-to-achieve werden sie häufig auch als Planungssprachen bezeichnet. Sie zeichnen sich durch eine methodische Orientierung bzw. Beschränkung der realisierbaren Modellstrukturen, meist in Form hierarchisch-additiv verknüpfbarer, tabellarischer Modelle (Spreadsheets) aus. Das Funktionsbündel schließt jedoch kommunikative Aspekte wie Textverarbeitung oder EMail aus.

Das gleiche gilt für eine zweite Gruppe von DSS-Generatoren, Expertensystemshells, die lediglich eine andere methodische Basis für Modellbildung sowie Funktionen der Auswertung, Analyse und Problemlösungsgenerierung aufweisen. Diese methodische Spezialisierung und Beschränkung erscheint sinnvoll, um einerseits nicht die Produktivitätsvorteile der Einzelkomponenten zu gefährden und andererseits ein notwendiges Maß an Überschaubarkeit und Handhabbarkeit der Werkzeuge zu bewahren.

Zur Rechnerunterstützung des Aufgabenspektrums ganzer Arbeitsplätze ist also eine Kombination mehrerer Anwendungen einer oder mehrerer Werkzeuge notwendig. Jedes Werkzeug verfügt dabei jedoch zumeist über individuelle Komponenten zur Gestaltung der Benutzer-

oberfläche. Die Zusammenführung an einem Arbeitsplatz erfordert hingegen eine einheitliche Oberfläche. Dies kann entweder dadurch erreicht werden, daß einem der kombinierten Werkzeuge diese Aufgabe übertragen wird, oder dadurch, daß ein drittes, ausschließlich oder schwerpunktmäßig auf Benutzerdialog spezialisiertes Werkzeug zum Einsatz kommt.

Zur Realisierung dieser einheitlichen Oberfläche können wiederum die Präsentations- und Dialogfunktionen von EIS-Werkzeugen eingesetzt werden, um auf diese Weise den ursprünglich für Führungskräfte konzipierten Bedienungskomfort allen Anwendergruppen zugänglich zu machen. Trotz der geschlossenen Konzeption einiger, zumeist anfänglicher Produkte verfügen sie in ihrem konstruktiven Ansatz prinzipiell und in jüngster Zeit auch in zunehmendem Maße über umfangreiche Schnittstellen zur Kommunikation mit anderen Anwendungen, die z.B. mit konventionellen oder wissensbasierten DSS-Generatoren erstellt wurden, aber auch sonstigen Werkzeugen der Bürokommunikation, wie Textverarbeitung oder EMail.

Aus diesen Gründen werden im Teil II die beiden Werkzeugklassen der EIS- und DSS-Generatoren, letztere getrennt nach konventioneller und wissensbasierter Methodenbasis, bezüglich ihrer klassenbildenden Modellierungs- und Entwicklungskonstrukte detailliert dargestellt und bezüglich ihres Leistungsspektrums analysiert. Die Konstrukte werden anhand konkreter Ausgestaltungsformen ausgewählter Produkte beispielhaft erläutert.

Teil II

Analyse standardisierter Entwicklungswerkzeuge

5. Generatoren für Executive Information Systems (EIS)

5.1. Referenzarchitektur

Als Basis für die Analyse und die Gestaltung von Unternehmens- und Datenmodellen sowie Softwaresystemen haben sich sogenannte Referenzmodelle bewährt (vgl. [Scheer 90, Wirtschaftsinformatik], S. 519ff.). Sie umfassen unabhängig von konkreten Anwendungsfällen alle relevanten strukturellen und funktionalen Elemente und Beziehungen für das zu untersuchende oder zu gestaltende Zielsystem. Im vorliegenden Fall werden Referenzmodelle zur Klassifizierung der Konstrukte von Entwicklungswerkzeugen entwickelt.

Abb. 5-1 zeigt das Referenzarchitekturmodell, das in allgemeiner Form alle maximal möglichen strukturellen und funktionalen Komponenten von EIS-Generatoren umfaßt. Wenngleich produktmäßig häufig verschmolzen, kann logisch in die beiden strukturellen Komponenten EIS-Datenbasis und EIS-Anwendung unterschieden werden. Sie bestehen jeweils aus einer Aufbau- und Ablaufkomponente:

- Unter *Aufbaukomponente* können in der Datenbasis das Datenmodell und die Datentypen, in der Anwendungsbasis das Layout von Bildschirmen oder Bildschirmfenstern verstanden werden.
- Mit *Ablaufkomponente* werden jeweils funktionale Elemente bezeichnet, in der Datenbasis z.B. die Spezifikation einer Updateprozedur incl. Beschreibung der Datenquelle sowie Selektionskriterien und Aufbereitungsberechnungen, in der Anwendungsbasis z.B. Interaktionsmöglichkeiten.

Untereinander bzw. zur Umwelt, d.h. in Bezug auf originäre interne und externe Informationsquellen sowie EIS-Entwickler und -Anwender, sind die beiden Komponenten durch Laufzeitsysteme verbunden, die jeweils die Interpretation und Umsetzung der funktionalen Spezifikationen übernehmen. Sie sind in der Regel nicht durch den Anwendungsentwickler modifizierbar.

Angebotene EIS-Werkzeuge umfassen nicht immer alle Komponenten des Referenzmodells. Vor allem neuere Produkte, deren Schwerpunkt auf einer integrativen Oberfläche zu verschiedenen Datenbasen oder Server-Applikationen liegt, verzichten auf eine eigene EIS-Datenbasis. Beispiele hierfür sind LightShip von Pilot Executive Software GmbH und Forest&Trees von Channel Computing, Inc.. Die Datenverwaltungsfunktion entfällt hier

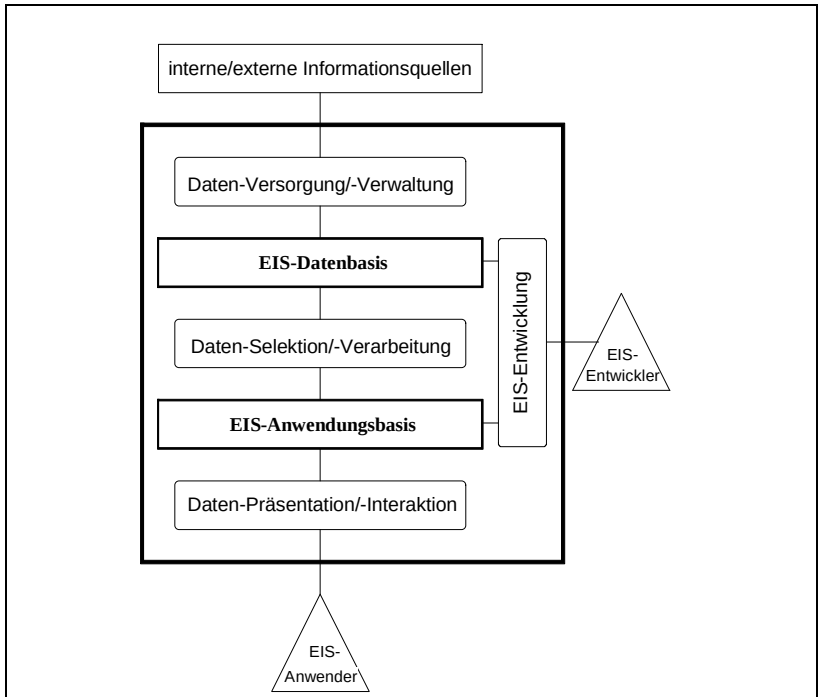


Abb. 5-1: Referenzarchitekturmodell für EIS-Generatoren

völlig und wird an das jeweilige externe Quellsoftwaresystem delegiert. Datenversorgung aus originären Quellen und -Selektion aus (sekundärer) EIS-Datenbasis verschmelzen, während Datenverarbeitung teilweise ebenfalls ausgelagert, teilweise aber auch im EIS angesiedelt werden können.

Frühere, klassische EIS-Werkzeuge, wie CommanderEIS von Comshare AG und PILOT von Pilot Executive Software GmbH verfügen hingegen über eine eigene Datenbasis. Während bei PILOT sowohl in der Datenversorgungs- als auch in der (sekundären) Datenselektionsphase arithmetische Informationsverarbeitungsfunktionen spezifiziert werden können, z.B. zur Berechnung von Summen, Abweichungen etc., ist bei CommanderEIS das Verarbeitungsspektrum auf einfache Formatierungen beschränkt, so daß auch hier trotz EIS-Datenbasis arithmetische Aufbereitungen außerhalb des Werkzeugs realisiert werden müssen. Dies hängt unmittelbar mit der verwendeten Struktur der Datenbasis zusammen. Aufgrund

dieser wechselseitigen Abhängigkeiten der strukturellen und funktionalen Elemente wird im folgenden die Darstellung der EIS-Konstrukte in die Phasen Informations-Verwaltung, -Selektion, -Präsentation und -Verarbeitung unterteilt.

5.2. Informations-Verwaltung

5.2.1. Klassifikation

Als Grundprinzipien für die Strukturierung, Speicherung und Verwaltung der EIS-Datenbasis können Dokumente, Datenbanken und Modellbanken unterschieden werden. Dabei spielt es zunächst keine Rolle, ob die EIS-Datenbasis integraler Bestandteil der EIS-Software ist oder extern per Schnittstelle realisiert ist. Im folgenden werden diese drei, häufig kombiniert eingesetzten Typen an konkreten EIS-Werkzeugen beschrieben. Dabei werden EIS-spezifische, zusätzliche Konstrukte hervorgehoben.

5.2.2. Dokumente

Dokument-orientierte Systeme stellen die klassische EIS-Datenbasis-Architektur dar. Darauf basierende Anwendungen werden hersteller-übergreifend als Briefing-Book, also eine Art elektronisches Berichtswesen, bezeichnet.

Dokumente stellen bereits weitestgehend präsentationsreif formatierte Daten, Texte und Graphiken dar, die zur Ausführungszeit nur noch in dafür vorgesehene Bildschirmfenster eingeblendet und allenfalls noch bezüglich Zeichengröße und Farbe oder Position und Ausschnitt manipuliert werden. Der Vorteil dieses Konzepts ist darin zu sehen, daß Informationen aus verschiedensten, bestehenden Informationssystemen, z. B. Großrechnerberichtswesen oder auch PC-Planungsmodellen, ohne großen Konvertierungsaufwand übernommen werden können. Hierzu verfügen die EIS-Werkzeuge über direkte Schnittstellen zu verschiedenen, gängigen Dateiformaten, z.B. ASCII, PCX, BMP, etc..

Wenngleich praktisch alle angebotenen EIS-Werkzeuge diesen Datenbasistyp unterstützen, sind in Bezug auf Verwaltung, Präsentation und Verarbeitung gravierende Unterschiede auszumachen. Als Maßstab soll das in diesem Bereich am weitesten entwickelte Produkt CommanderEIS der Comshare AG näher betrachtet werden:

CommanderEIS verfügt über eine eigene Bibliotheksverwaltungskomponente, in der jedes Dokument mit den Attributen EIS-interner Name, Typ und Eigner sowie Datum und Uhrzeit des letzten Update erfaßt wird (vgl. Abb. 5-2). Als Typen im engeren Sinne stehen REPORT

für Textdateien im Sinne von ASCII-Files und GRAPH für gängige Graphikformate externer PC-Softwarepakete zur Auswahl. Die eigentlichen Daten können wahlweise als Kopie in die Bibliothek fest aufgenommen oder via Link zu einem DOS-Dateinamen auf dem lokalen PC-Arbeitsplatz oder einem Dateiserver in einem PC-Netzwerk definiert werden. Das Eigner-Attribut dient zur Steuerung von Zugriffsberechtigungen auf das Dokument, kann aber auch zur Versionsverwaltung verwendet werden.

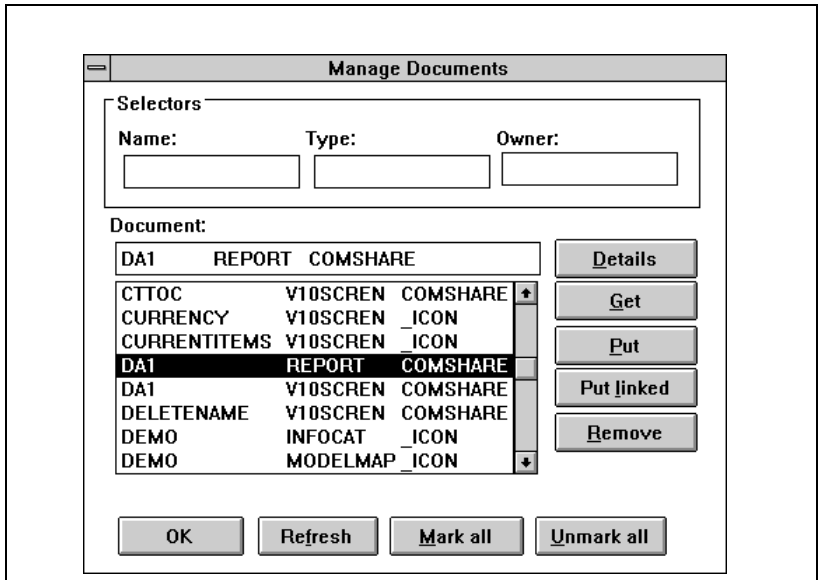


Abb. 5-2: Dokumentverwaltung in CommanderEIS

Die Dokumentbibliothek kann lokal auf der PC-Workstation oder dezentral auf einem Dateiserver in einem PC-Netzwerk für den gleichzeitigen Zugriff durch mehrere Anwender angesiedelt werden. Für größere, kommerzielle Anwendungen steht zusätzlich eine Großrechnerbasierte Komponente zur Verfügung, in der unternehmens- oder abteilungsweit die Dokumente aller Anwender redundanzfrei gesammelt und zu dokumentspezifischen Stichzeitpunkten auf die lokalen Dokumentbibliotheken heruntergeladen werden können. Für dieses Downloading stehen spezielle, automatisierbare Administrationskommandos zur Verfügung, die u.a. auch ein selektives, ereignisorientiertes Update gestatten, z.B. in Abhängigkeit

vom Datumsattribut zwischen zentralem und dezentralem Dokument oder in Bezug auf das Eignerattribut, das in der Großrechnerbibliothek mit der dortigen Userid gleichzusetzen ist. Ein anfangs möglicher Direktzugriff des Anwenders auf das zentrale Dokument in der Großrechnerbibliothek wurde aus Gründen der Zugriffszeiten wieder abgeschafft.

Neben den beschriebenen Attributen Name, Typ, Eigner, Datum und Link umfaßt das Dokumentkonstrukt weitere funktionale Eigenschaften. Über den Dokumenttyp werden nämlich bereits Formen und Eigenschaften der Informationspräsentation festgelegt, z.B. die Dokumentinterpretation bei der Anzeige in EIS-Bildschirmen oder die automatische Aktivierung von Richtungspfeilen zum Rollen von Textdokumenten in Fensterausschnitten.

Neben den Dokumenttypen im engeren Sinne erstreckt sich die Dokumentverwaltung in CommanderEIS auf die gesamte Anwendungsspezifikation, d.h. es existieren Dokumenttypen für die Struktur und Funktionalität der EIS-Oberfläche. In Bezug auf das Referenzmodell bilden EIS-Daten- und -Anwendungsbasis in CommanderEIS also eine Einheit. Diese Dokumenttypen sind im wesentlichen V10SCREN für einzelne Bildschirme, SCRIPT für Benutzer- oder Ereignis-gesteuerte Anwendungsfunktionen, DOS und COMMAND für DOS- oder CommanderEIS-Kommandosequenzen sowie PERIODIC für zeitgebundene, automatische Aktionen. Auf diese Elemente wird unter den Punkten 5.4.2 und 5.4.3 näher eingegangen.

Für die EIS-Datenbasis relevant sind noch die Dokumenttypen INFOCAT und MODELMAP. Sie realisieren eine Schnittstellenspezifikation zu rein numerischen Daten, die in externen Modellen der Planungssprache OneUp desselben Anbieters (außerhalb des EIS) gespeichert sind. Der Dokumenttyp INFOCAT beschreibt dabei die potentiell selektierbare Datenteilmenge des OneUp-Modells, ergänzt um eine Standard-Einstiegsansicht. Über Parameter ist es insbesondere möglich, im Sinne der Zugriffssteuerung Richtung und Tiefe des Drill-Down für einzelne Anwender zu differenzieren, so daß Teilsichten auf ein integratives Gesamtmodell realisiert werden können. Der Dokumenttyp MODELMAP ist lediglich die kompilierte Form dieser View-Spezifikation, verhindert Manipulationen der Anwender am definierten Modellausschnitt und bewirkt bei Aufruf durch den Anwender, daß die entsprechende Sicht auf das spezifizierte Modell hergestellt und visualisiert wird. Leider sind Aufbau und Funktionalität der Oberfläche dieser Teilanwendung nicht durch den Entwickler modifizierbar.

Das beschriebene Dokumentkonzept weist typische Züge objektorientierter Systeme auf, indem Daten und Funktionen zu Objektklassen zusammengefaßt bereitgestellt werden, aus denen der EIS-Entwickler per Instanzenbildung Anwendungen konfigurieren kann. Dennoch

kann es nicht als objektorientiertes Werkzeug charakterisiert werden, da die Klassen weder attribut- oder methodenmäßig verändert oder erweitert, noch Unterklassen mit Vererbungseigenschaften gebildet werden können.

Alle anderen EIS-Werkzeuge beschränken sich in ihren Dokument-orientierten Komponenten darauf, im Rahmen der Anwendungsspezifikation, d.h. der Festlegung der Anwendungsoberfläche, DOS-Dateinamen als Datenschnittstelle zuzulassen. Es findet also weder eine physische noch eine logische Datenverwaltung statt, insbesondere fehlen jegliche Voraussetzungen und Mechanismen für Zugriffsteuerung und Datenschutz.

5.2.3. Datenbank

Im Gegensatz zum Dokumentenkonzept weist die datenbank-orientierte Informationsverwaltung weder eine präsentationsgerechte Formatierung noch Selektion auf. Die Daten sind nach sachlogischen Gesichtspunkten grundsätzlich anwendungsneutral abgelegt. In den meisten Fällen beschränken sich EIS-Werkzeuge mit diesem Datenbasistyp darauf, standardisierte Schnittstellen zu gängigen PC- und Großrechnerdatenbanksystemen anzubieten, z.B. in Form von SQL oder einer SQL-ähnlichen Abfragesprache. Aus Gründen notwendig kurzer Antwortzeiten kann dabei in den wenigsten Fällen direkt auf die originären Datenbanken zugegriffen werden, vielmehr muß mit Hilfe eines externen, in der Regel relationalen Datenbanksystems eine externe EIS-Datenbasis aufgebaut werden. Um die spezifischen Abfragefunktionen von EIS optimal zu versorgen, bietet es sich an, über das normale Maß relationaler Datenbanksysteme hinausgehende Datentypen und Funktionen vorzusehen. Beispiele für solche erweiterte Konstrukte sind nicht nur in den wenigen EIS-Werkzeugen mit eigener datenbankorientierter Datenbasisverwaltung, wie z.B. dem Produkt PILOT von Pilot Executive Software GmbH, sondern zunehmend auch in DSS-Generatoren zu finden, deren Modelle auf einer integrierten Datenbank basieren, wie z.B. NOMAD von MUST Software oder macControl von Breitschwerdt&Partner.

Analyse- und Planungsaufgaben basieren häufig auf Zeitvergleichen. Deshalb ist eine effiziente und komfortable Abbildung von Zeitreihen sowohl für Planungssprachen und DSS-Generatoren als auch mehr noch für Führungsinformationssysteme von großer Bedeutung. In Datenbanksystemen bietet sich das Konstrukt eines elementaren Datentyps Zeitreihe an. Charakteristika des Datentyps Zeitreihe sind ein Start- und Endtermin im Datumsformat sowie eine Periodizität, z.B. täglich, wöchentlich, monatlich, jährlich usw.. Hinzu kommt meist noch eine textuelle Beschreibung der Zeitreihe. Nach diesem Prinzip arbeiten NOMAD und PILOT (vgl. Abb. 5-3). Hinzukommen müssen aber noch zeitreihenspezifische Funktionen zur automatisch synchronisierten Aggregation heterogener Zeitreihen,


```
PILOT-table-Definition:
...
MAKE TABLE Kennzahlen
MAKE FIELD Umsatz TIMESERIES
...
MAKE RECORD FIELD Umsatz
  TSLIST(10,12,14,16) START 1/92 QUARTERLY
...

NOMAD-table-Definition
...
MASTER Kennzahlen;
ITEM Umsatz SDATE'1/92' QUARTERLY
...
```

Abb. 5-3:	Beispiele für Datentyp Zeitreihe

einem typischen Anwendungsfall bei EIS im Zusammenhang mit dem Drill-Down: sollen bspw. monatlich vorliegende Kostenwerte mit täglich eingehenden Umsatzwerten verglichen werden, muß die Zeitreihenarithmetik in Verbindung mit einem integrierten, unendlichen Kalender automatisch dafür sorgen, daß die Tageswerte zu Monatswerten aggregiert werden, bevor die Umsatz-Kosten-Differenz gebildet wird.

Eine alternative Lösung zum eigenständigen Datentyp Zeitreihe wird im DSS-Generator macControl von Breitschwerdt & Partner gewählt. Diese Datenbasis kann als mehrdimensionale Datenbank charakterisiert werden. Hierbei wird Datenwerten ex ante kein fester Typ zugeordnet, vielmehr bestimmt sich dieser aus dem eingegebenen Datenwert automatisch. Datenwerte können aber durch beliebige Merkmale klassifiziert werden, die ihrerseits zu Merkmalsgruppen zusammengefaßt werden können. Üblicherweise stellen die Merkmalsgruppen Dimensionen wie Produkt, Region, Version usw. dar. Die Zeit ist in diesem Konzept nicht als Datentyp sondern als besondere, herausgehobene, integrierte und vordefinierte Merkmalsgruppe realisiert.

Der Vorteil dieses Konzepts ergibt sich hauptsächlich im Zusammenhang mit Modellen, die auf und in dieser Datenbasis definiert werden können. Er ist darin zu sehen, daß Informationen unterschiedlichster Dimensionen und Zeithorizonte parallel in einem Modell gehalten werden können, ein in der Praxis häufig auftretendes Problem, das mit den meisten modellbasierten Planungsssprachen und DSS-Generatoren nur ineffizient zu lösen ist - wie die Analyse in Kapitel 6.2.1.1 zeigen wird. Beispiele dafür sind Mehrproduktunternehmen,

deren Produkte in unterschiedlichen Regionen oder Vertriebskanälen abgesetzt oder deren Umsätze zeitlich verschieden überwacht werden und die daher individuell dimensioniert werden müssen, aber dennoch auf einem gemeinsamen (monatlichen) Nenner im Berichtswesen zusammengefaßt werden sowie individuell hinterfragbar bleiben sollen.

Weitere Beispiele für potentielle Konstrukte in der Informationsverwaltung von EIS sind Datenattribute, die unmittelbar für die Formatierung der Präsentation genutzt werden können. Hierzu gehört bspw. die Dimension numerischer Werte, z.B. als Mengen-, Gewichts- oder Währungseinheit. Bspw. ist in der PILOT-Datenbank ein solches Attribut je Tabellenfeld vorgesehen. Im diesem Falle kann dann funktional für die Aggregation eine automatische Umrechnung realisiert werden, z.B. von kg und Tonnen in kg oder bei unterschiedlichen Währungen durch Hinzuziehen von zentral gespeicherten Wechselkursen in eine beliebig wählbare Abrechnungseinheit. Ferner können die Dimensionen komplexer Kennzahlberechnungen automatisch generiert werden. Schließlich kann so der Formatierungsaufwand bei der Präsentation im EIS in Form automatisch generierter Dimensions-Präfixe oder Suffixe erheblich reduziert werden.

5.2.4. Modelle

Modelle als Informationstyp für EIS-Datenbasen sind dem Datenbanktyp sehr verwandt, verfügen aber im Gegensatz zu diesen über die Abbildungsmöglichkeit von rechnerischen Beziehungen in Form von Definitions- oder Verhaltensgleichungen, deren Komplexität und Vernetzung weit über das im Rahmen relationaler Datenbankverknüpfungen Mögliche hinausgeht. Dafür sind sie zumeist auf numerische Datentypen beschränkt. Spezielle Konstrukte zur Erhöhung von Entwicklungsproduktivität und Modellierungskomplexität sind Mehrdimensionalität, Nichtprozeduralität und Variablentypen, auf die im Kapitel 6.2 im Zusammenhang mit DSS-Generatoren und Planungssprachen detailliert eingegangen wird, da sie zumeist außerhalb der EIS-Datenbasis verwaltet werden. EIS-Generatoren beschränken sich vielmehr auf die Spezifikation von Selektionsschnittstellen zu diesen Modellen. Eine Ausnahme bildet das Produkt macControl, bei dem die Modellbildung in Form von Datenbasis-Funktionen integriert sein kann.

5.3. Informations-Selektion

5.3.1. Klassifikation

Wichtigstes Element von EIS-Generatoren sind Mechanismen der Informationsselektion zum Zwecke der Reduktion der Informationsflut. Bezogen auf das Referenzmodell (vgl. Kapitel 5.2.1) können zwei zeitlich getrennte Typen unterschieden werden:

- Reduktion der Informationsmenge bei der Datenübernahme aus internen/externen Quellen in die EIS-Datenbasis und
- Reduktion der Informationsmenge und -komplexität im Rahmen der EIS-Anwendung, d.h. beim Zugriff auf die EIS-Datenbasis, egal ob diese inner- oder außerhalb des EIS-Werkzeugs realisiert ist.

Im ersten Fall wird das insgesamt durch den EIS-Anwender abrufbare Informationsvolumen festgelegt. Es kann in der Regel nur durch Einschaltung des EIS-Entwicklers modifiziert werden. Im zweiten Fall können solche Spielräume für den EIS-Anwender eingebaut werden, wobei deren Grad durch den Typ der EIS-Datenbasis determiniert wird. Weiterhin ist nach dem Akteur der Selektion zu unterscheiden, d.h. ob und in welchem Ausmaß EIS-Entwickler oder auch EIS-Anwender Grad und Richtung der Selektion bestimmen können. Somit ergibt sich das in Abb. 5-4 dargestellte Klassifizierungsschema mit den dort eingetragenen Möglichkeiten, Restriktionen und Konsequenzen.

Die Informationsselektion wird bei allen EIS-Werkzeugen durch spezielle Konstrukte der Anwendungsspezifikation festgelegt: Drill-Down und Exception-Reporting. Im Falle Datenbank- und Modell-orientierter Informationsverwaltung wird das Spektrum um die bereits in den Kapiteln 5.2.3 und 5.2.4 angeführten Mechanismen der Abfragespezifikationen erweitert. Bei Dokument-orientierten Anwendungen bezieht sich die Abfragespezifikation jeweils auf das gesamte Dokument, lediglich der für den Anwender sichtbare Ausschnitt kann eingeschränkt werden. Während bei LightShip jede beliebige Teilmenge des Dokuments festgelegt und dem EIS-Anwender zusätzlich innerhalb dieses Ausschnitts das horizontale und vertikale Rollen ermöglicht werden kann, besteht bei CommanderEIS nur die Wahl zwischen den Alternativen eines beliebigen, aber festen Ausschnitts oder eines flexiblen, dann aber auf das gesamte Dokument bezogenen Zugriffs.

5.3.2. Abfragespezifikationen

Hierbei ist zu unterscheiden, ob das EIS-Werkzeug über eine eigene, häufig am SQL-Stan-

		Akteur der Informations-Selektion	
		EIS-Entwickler	**EIS-Anwender**
Typ der EIS-Datenbasis	**Dokumente**	Selektion des gesamten Dokuments; Festlegung des sichtbaren Ausschnitts; Festlegung vertikaler und/horizontaler Rollmöglichkeiten	vertikales/horizontales Rollen im Rahmen der Vorgaben
	Datenbank	Tabellen-/Feld-Auswahl; Berechnungsfunktionen (MIN, AVG,...); Auswahlbedingungen (WHERE,...); Gruppierungen (GROUP); Parametrisierung der Abfrage	Auswahl im Rahmen der vorgegebenen Parametrisierung
	Modell	Variablen-/Dimensions-Auswahl; zusätzliche Gruppierungen	Definition individueller Teilsichten im Rahmen der vorgegebenen Gesamtsicht

Abb. 5-4: Klassifikationsschema der Informationsselektion in EIS-Generatoren

dard orientierte oder auf Modelle bestimmter, alternativer DSS-Generatoren ausgerichtete Abfragesprache verfügt oder lediglich die Abfragesprachen der (externen) Quellsysteme über eine Kommunikationsschnittstelle benutzt.

Ein Beispiel für den letzten Fall stellt die Verwendung des dynamischen Datenaustauschs (DDE) unter MS-Windows durch die Produkte LightShip oder Forest&Trees dar. Zwar sind dadurch alle ebenfalls DDE-fähigen Softwaresysteme unter MS-Windows unterschiedlichster methodischer Art, wie z.B. das Tabellenkalkulationsprogramm MS-Excel, die Datenbanksysteme SQL-Windows oder SuperBase , die Planungssprache CA-Compete oder die Expertensystemshell KappaPC erreichbar, der EIS-Entwickler muß jedoch jeweils die unterschiedliche Syntax der Zielsysteme beherrschen (vgl. Abb. 5-5).

Ähnlich sieht es in allen Fällen aus, in denen die Informationsquelle durch Modelle von DSS-Generatoren gebildet wird. Aber auch hier sind verschieden komfortable Formen zu unterscheiden, und zwar ob überhaupt eine Art Abfragesprache existiert, oder ob lediglich auf dem Niveau dokumentorientierter Systeme eine Dialogsitzung mit dem DSS-Generator simuliert und gescannt wird, was prinzipiell mit dem o.g. Datenaustausch mittels DDE ver-

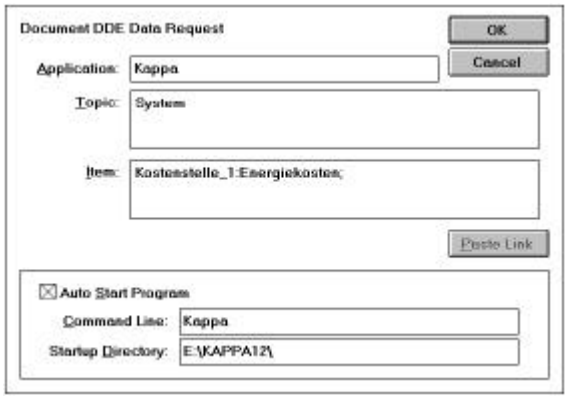


Abb. 5-5: Beispiel der Informationsselektion per DDE mit LightShip

gleichbar ist. Ein Beispiel für diesen letzten Fall stellte das bis 1991 vertriebene System ExecutiveEdge von EXECUCOM dar (vgl. [Rieger 90, EIS-Generatoren]).

Ein Beispiel für eine spezielle Abfragesprache auf DSS-Modelle ist in CommanderEIS mit den Dokumenttypen INFOCAT und MODELMAP realisiert. Abb. 5-6 zeigt ein Beispiel einer solchen selektiven Abfragespezifikation, die für den vollen Umfang der Modellstruktur automatisch generiert und anschließend durch den EIS-Entwickler per Text-Editor modifiziert werden kann. Dabei sind sowohl Einschränkungen der Sicht durch Löschen einzelner Dimensionsausprägungen als auch Erweiterungen in Form neuer Gruppierungen elementarer Dimensionsausprägungen möglich. Diese Gruppen bewirken ausschließlich additive Berechnungen, was im Falle von Kennzahlen zu unsinnigen Ergebnissen führt. Im übrigen werden keinerlei Modellberechnungen induziert, d.h. das abgefragte Modell muß vollständig berechnet sein. Die abgefragten bzw. neudefinierten Gruppenhierarchien bilden die Basis für ein modellbasiertes Drill-Down, das im nächsten Kapitel detailliert behandelt wird.

Ein Beispiel für eine integrative, eigene Abfragesprache stellt das EIS-Werkzeug Forest&Trees von Channel Computing dar. Dadurch entfällt das Problem, die unterschiedliche Syntax verschiedener Quellsysteme beherrschen zu müssen. Das Spektrum uniform ansprechbarer Datenquellen reicht von Datenbanksystemen, z.B. R:Base, dBase, PARADOX usw. über Tabellenkalkulationssysteme, z.B. Lotus-123 oder MS-Excel, bis hin zu ASCII-

```
Model: "D:\ONEUP\EKG.OU1"
Version: MAP1
Date: "19:40 Uhr am Freitag, den 3.2.1989"
Format: 8,2
Expand: UP
Category:  Produkte "Artikel-Hierarchie"
    AG1           Detail        " Art.gr.1"
    AG2           Detail        " Art.gr.2"
    AG3           Detail        " Art.gr.3"
    GB1           Consolidate   "Gesch.ber.1"
    AG4           Detail        " Art.gr.4"
    AG5           Detail        " Art.gr.5"
    GB2           Consolidate   "Gesch.ber.2"
    TOTAL         Consolidate   "Total"
groups:
    GB1 = AG1,AG2,AG3
    GB2 = AG4,AG5
    TOTAL = GB1,GB2
display:
    *   GB1
    *   GB2
  ** TOTAL
Category:  Period  "Periode"
    P1    Detail           " A:Budget 1985"
    P2    Detail           " A:Jan-Plan"
    P3    Detail           " A:Ist"
    P4    Detail           " B:Budget 1985"
    P5    Detail           " B:Jan-Plan"
    P6    Detail           " B:Mai-Plan"
    P7    Detail           " C:Budget 1985"
    P8    Detail           " C:Jan-Plan"
    P9    Detail           " C:Mai-Plan"
    P25   Consolidate      "Budget 1985"
    P26   Consolidate      "Jan-Plan"
    P27   Consolidate      "Mai-Plan"
groups:
    P25 = P1 P4 P7
    P26 = P2 P5 P8
    P27 = P3 P6 P9
```

```
Category: Variable "Kennzahlen"
    ANFBEST   Detail       " Anf.bestand"
    ANFBESTWV Detail       " Anf.best.WV"
    MENGE     Detail       " Menge"
    ZUG       Detail       " Zug„nge"
    FP        Detail       " Fakturpreis"
    VP        Detail       "
Verrechn.preis"
display:
    **    MENGE
     *    ANFBEST
     *    ANFBESTWV
     *    ZUG
     *    FP
     *    VP
Category: Kunde "Kunde"
    R01     Detail        "  1. Kunde"
    R02     Detail        "  2. Kunde"
    R06     Detail        "  Restkunden"
    R07     Consolidate   "alle Kunden"
groups:
   R07 = R01,R02,R06
display:
  ** + R07
```

Abb. 5-6:	Informationsselektion aus DSS-Modellen am Beispiel von Commander-EIS

Textdateien, die hierzu im Rahmen einer einmaligen Vorverarbeitung "gerastert" werden müssen. Dabei können in Analogie zu SQL Berechnungsfunktionen, z.B. Minimum, Maximum, Anzahl oder Durchschnitt sowie Bedingungs- und Gruppierungsklauseln (WHERE, GROUP, HAVING) vom EIS-Entwickler eingefügt werden. Da es möglich ist, die Abfragesequenzen mit Parametern zu versehen, kann die Selektion auch dem EIS-Anwender zur Verfügung gestellt werden. Abb. 5-7 zeigt ein Beispiel für eine Abfragespezifikation auf ein als dBase-Tabelle abgespeichertes CA-Compete-Modell. Dabei sind für den Anwendungsentwickler keinerlei Kenntnisse über die Tabellen- und Feldstrukturen der Quellsysteme notwendig, da die gesamte Abfragespezifikation wahlweise auch Menü- und Masken-gesteuert durchgeführt werden kann, d.h. alle selektierbaren Tabellen und Felder sowie Berechnungsfunktionen sowie Bedingungs- und Gruppierungsklauseln von der Entwicklungsoberfläche von Forest&Trees zur Auswahl angeboten werden.

Bei Forest&Trees werden die Abfrageergebnisse physischer Bestandteil der Anwendung, d.h. die selektierten Daten werden als Kopie im EIS gespeichert. Zwar können sie durch Sekundärabfragen in derselben oder auch von anderen EIS-Anwendungen weiterverwendet

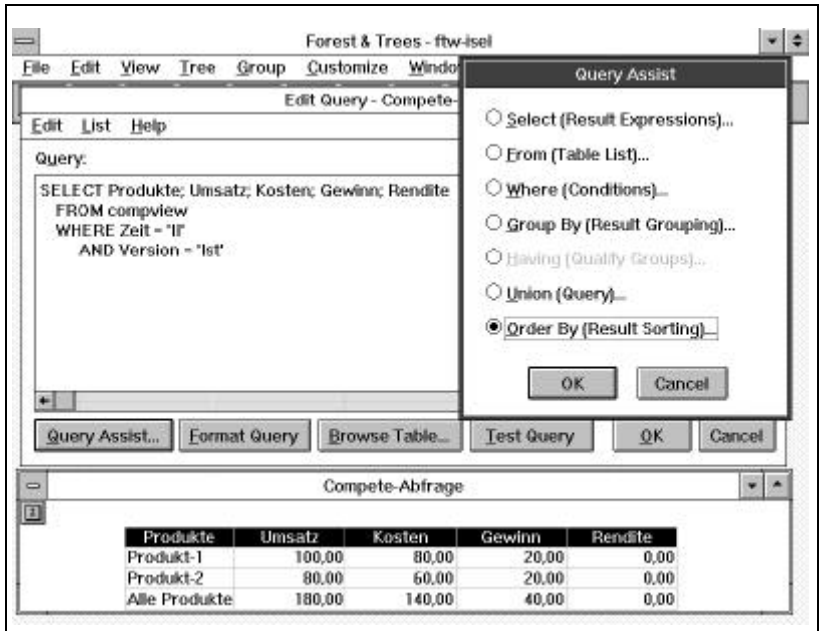


Abb. 5-7:	Informationsselektion auf Basis einer integrativen Abfragesprache am Beispiel von Forest&Trees

werden, es entstehen jedoch nicht unerhebliche Synchronisationsprobleme im Falle des Updates der originären Datenbasis (vgl. Kapitel 5.3.3).

Dieses Problem wird durch eine zweistufige Vorgehensweise beim Datenbankschnittstellen-Konstrukt LENS des EIS-Generators LightShip vermieden. Im ersten Schritt werden hier aus der originären Datenbanktabelle zu selektierende Felder (LoadFields...) und Datensätze (LoadConditions...) spezifiziert. Das Ergebnis kann in einer Cache-Datei, unabhängig von einer bestimmten EIS-Anwendung, abgespeichert werden. Im zweiten Schritt können dann unterschiedliche Anwendungsteile oder auch Anwendungen weitere Feld- (Results...) und Satz-spezifische (Conditions...) Restriktionen auf diese Cache-Datei vornehmen. Da Änderungen der Cache-Dateien automatisch in den Anwendungen aktualisiert werden, entsteht kein Zusatzaufwand in puncto Synchronisation, d.h. der EIS-Entwickler muß nicht explizit dafür sorgen, daß alle Sekundärabfragen im Falle von Änderungen der Primärabfrage (Cache-Datei) aktualisiert werden.

Ähnlich wie Forest&Trees verfügt auch LightShip-LENS über eine komfortable Oberfläche zur Abfragespezifikation (vgl. Abb. 5-8). Leider sind die Bedingungsdefinitionen nur rudimentär realisiert. So können ausschließlich UND-verknüpfte Bedingungen für nicht-numerische Datenbankfelder spezifiziert werden. Auch rechnerische Feldverknüpfungen sind (derzeit) nicht möglich. Hierfür muß auf die - allerdings nicht per Menüauswahl unterstützte - SQL-Option zurückgegriffen werden. Zwar können in beiden Fällen Parametrisierungen der Abfragespezifikation durch Variablen vorgenommen werden, diese schließt im SQL-Fall incl. Cache-Datei allerdings leider den Update-Automatismus aus. Somit lassen sich wertabhängige Abfragen, wie sie bspw. beim selektiven Exception-Reporting dringend benötigt werden, nur durch redundantes und zeitaufwendiges Laden der originären Datenbasis realisieren.

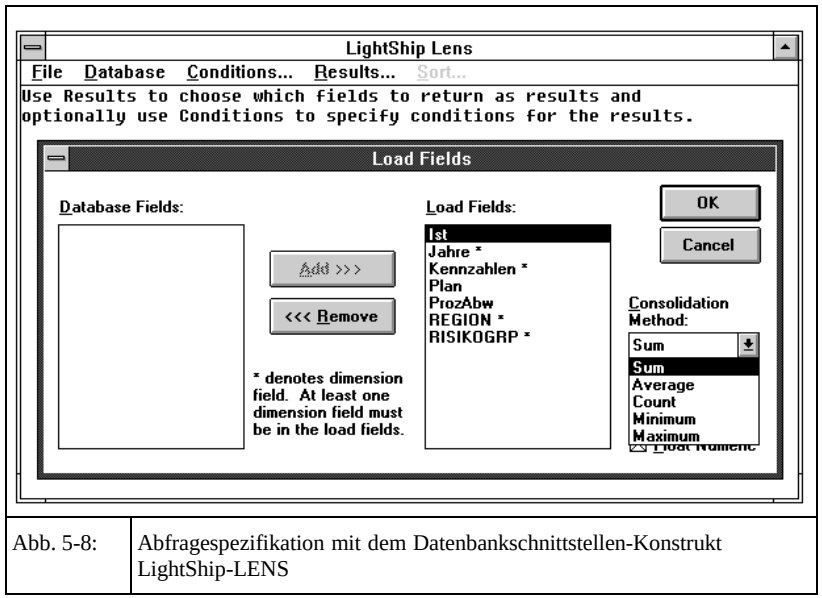


Abb. 5-8:	Abfragespezifikation mit dem Datenbankschnittstellen-Konstrukt LightShip-LENS

Dafür ist in LightShip-LENS mit der automatischen Dimensionserkennung ein effizientes Konstrukt im Hinblick auf die Drill-Down-Funktionalität von EIS-Anwendungen integriert. LENS interpretiert dabei alle nicht-numerischen Felder als (potentielle) Drill-Down-Dimensionen. Numerische Felder werden beim Laden automatisch über identische Dimensionskombinationen aggregiert, wobei zwischen Summierung, Durchschnitt, Minimum oder

Maximum gewählt werden kann. In Verbindung mit Variablen lassen sich so EIS-typische Datenbankabfragen auf unterschiedlichen Aggregationsniveaus äußerst effizient spezifizieren.

5.3.3. Update

Die Wartungseffizienz von EIS-Anwendungen wird in erheblichem Maße von Konstrukten zur Spezifikation des automatischen Update der EIS-Datenbasis determiniert. Im Falle Dokument-orientierter Systeme mit eigener EIS-Datenbasisverwaltung - wie bei CommanderEIS - wird mit Download-Prozeduren gearbeitet, die zu vom EIS-Entwickler festlegbaren Stichzeitpunkten eine Aktualisierung auf Basis der Datumsattribute der Dokumente vornehmen. Sekundäreffekte können mangels Verknüpfungsmöglichkeiten von Dokumenten gleichen Typs in der EIS-Anwendung nicht auftreten sondern müssen im Vorfeld durch synchrone Aktualisierung sich inhaltlich überschneidender Dokumente in der Bibliothek verhindert werden. Beispiele hierfür sind etwa unterschiedlich sortierte, aggregierte oder "gekippte" Tabellen-Dokumente identischer Kennzahlen. Eine Konsistenz-sichernde Unterstützung ist damit nicht realisierbar.

Forest&Trees verfolgt demgegenüber eine Objekt-individuelle Spezifikation des Update. Abb. 5-9 zeigt die verfügbaren Aktualisierungsoptionen, die in Abhängigkeit vom Objekttyp aktiviert werden können. Für Datenbankabfragen stehen so unterschiedliche Stichzeitpunkte, für DDE-Verbindungen zusätzlich eine ereignisorientierte Option (DDE-Hotlink) zur Auswahl. Sekundäreffekte können allerdings automatisch nur für rechnerische Verknüpfungen realisiert werden. Bei Sekundärabfragen muß der Entwickler selbst dafür sorgen, daß diese aktualisiert werden, eine aufgrund der sehr komplizierten Spezifikationsweise in Form von Hilfsobjekten nicht nur aufwendige sondern höchst fehlerträchtige Lösung.

5.3.4. Drill-Down

Unter Drill-Down werden alle Dialogformen verstanden, die es dem EIS-Anwender gestatten, den Detaillierungsgrad abgerufener Informationen zu variieren, also z.B. von einer Darstellung von Summenwerten über die Zeit-, Produkt- oder Regions-Dimension zur Darstellung der zugehörigen Einzelwerte zu gelangen, die ihrerseits natürlich wieder Summenwerte sein können. Diese hierarchisch-additive Verknüpfung einer Drill-Down-Informationskette stellt aber nur einen, wenn auch häufig verwendeten, Spezialfall dar. Grundsätzlich müssen alle beliebigen Informationen der EIS-Datenbasis per Drill-Down verknüpfbar sein. So können Hintergrundinformationen zu einem kurzen erläuternden Kommentar bzw. einer standardisierten Situationsbewertung oder -einschätzung ("unkritisch" ... "Handlungsbedarf")

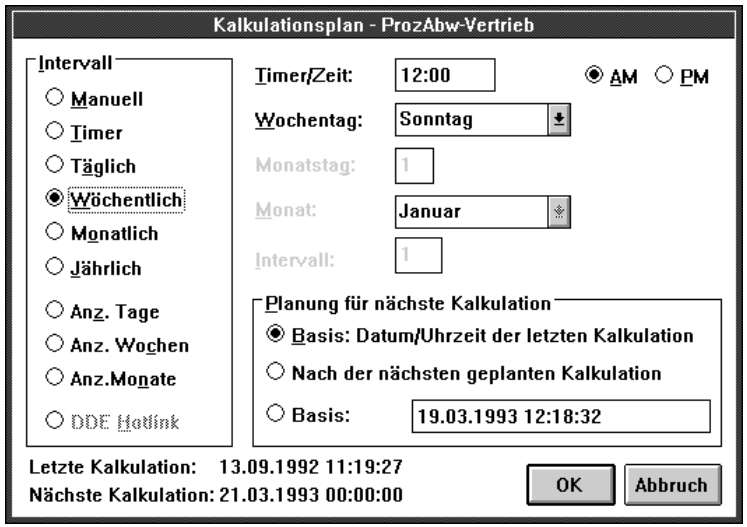


Abb. 5-9:	Update-Spezifikation in Forest&Trees

potentielle Ausgangspunkte für ein Drill-Down darstellen. Diese Hintergrundinformationen, und damit jede tiefere Drill-Down-Stufe, müssen grundsätzlich jede beliebige Komplexität und Form der EIS-Oberfläche aufweisen können, also Text, Daten, Grafik, Bilder und Aktionsfelder in beliebiger Kombination enthalten können.

Die Festlegung der Drill-Down-Verknüpfungen stellt einen Spezialfall der Verknüpfung von Bildschirmen einer EIS-Anwendung dar. Wie dies geschieht, hat erhebliche Auswirkungen auf die Flexibilität der EIS-Anwendung, determiniert in hohem Maße deren Wartungsaufwand und steht in direktem Zusammenhang mit dem Stichwort "datengetrieben" sowie dem Typ der zugrundeliegenden EIS-Datenbasis. Es können folgende Fälle unterschieden werden:

- *Spezifikation in der EIS-Anwendung*

  Üblicherweise werden EIS-Anwendungen in Form benannter Spezifikationen ganzer Bildschirmeinheiten entwickelt. Sprünge zwischen diesen Bildschirmen werden durch Aktionsfelder (Buttons, Hotspots) realisiert, die als Parameter den Namen des Zielbildschirms erhalten. Bei einer Drill-Down-Verknüpfung ist es in der Regel erwünscht, diesen Parameter dynamisch aus den z.B. in einer Tabelle angezeigten Informationen zu übernehmen oder abzuleiten.

Kennzeichnendes Merkmal des Drill-Down ist, daß der Ebenenwechsel durch den EIS-Anwender "intuitiv" gesteuert werden kann, indem die Aktion in der Regel durch "Anklicken" des zu detaillierenden, am Bildschirm angezeigten Elements ausgelöst werden kann. Wird bspw. in einem EIS-Bildschirm eine Liste mit Produktnamen und zugehörigen Umsatzzahlen dargestellt, so soll ein Anklicken jedes beliebigen Produktnamens der (variablen) Liste einen Detail-Bildschirm für genau dieses Produkt abrufen. Hierzu ist es notwendig, daß der EIS-Generator in der Lage ist, den selektierten Wert zu lesen und als Parameter für den Aufruf des Detailbildschirms zu übergeben. Da die Detailbildschirme häufig die gleiche Struktur aufweisen werden, genügt es nicht, wenn der Parameter den Bildschirmnamen bezeichnet, vielmehr muß er auch als Parameter für einen Musterbildschirm verwendet werden können. Schließlich wird der aus der Selektionsoperation unmittelbar zu lesende Wert nicht unmittelbar als Parameter verwendbar sein. So könnte in o.g. Beispiel die angezeigte Liste sowohl Umsatz- als auch Kostensummen je Produkt enthalten. Soll nun der Drill-Down nach Umsatz und Kosten differenzieren, so bietet sich jeweils als Drill-Down-Auslöser ein Anklicken der betreffenden Zahl an. Damit muß der EIS-Generator auch die Möglichkeit bieten, den benötigten Parameter für den Aufruf des Drill-Down-Bildschirms durch Match-Operationen zu bestimmen.

Erst die Existenz des beschriebenen Funktionsumfangs gewährleistet ein effizientes Drill-Down sowohl für den EIS-Anwender als auch besonders für den EIS-Entwickler. Es ist in dieser Form bspw. in den Produkten LightShip und Pilot realisiert, die dazu dem EIS-Entwickler bei der Definition von sog. Actions für Hotspots Funktionen wie MENULINE, MENUCOLUMN oder MENUTEXT anbieten. Der datengetriebene Drill-Down funktioniert dabei unabhängig von der verwendeten EIS-Datenbasis, also auch für Dokumente. Erschwerend kommt hierbei jedoch hinzu, daß die Identifikation des richtigen Dokuments nur über dessen Namen möglich ist. In diesem müssen also Identifikationsmerkmale "verschlüsselt" und über geeignete String-Operationen gematcht werden. Dadurch entsteht nicht nur erheblicher (kryptischer) Wartungsaufwand in Form von in der Anwendungsspezifikation zu codierenden Umsetzungstabellen, auch eine wirklich datengetriebene Anpassung ist praktisch ausgeschlossen. Typisches Beispiel hierfür ist das Produkt CommanderEIS.

- *Spezifikation in der EIS-Datenbasis*

  Mit dem oben beschriebenen datengetriebenen Drill-Down werden die Parameter praktisch durch die dargestellte Information, d.h. Inhalte der EIS-Datenbasis bestimmt. Dabei wurden Teile der dargestellten Information, im Falle einer Datenbankbasis wären

das etwa Feldinhalte des aktuell selektierten Datensatzes, direkt als Sprungparameter, d.h. für die folgende Abfragespezifikation, verwendet. Da EIS-Anwendungen sich häufig durch Aggregationen über mehrere Dimensionen auszeichnen, z.B. über Produkte, Regionen und die Zeit gleichzeitig, führt diese direkte Festlegung der Detaillierungsrichtung im Rahmen der EIS-Anwendungsspezifikation zu inflexiblen Aufrufhierarchien bzw. redundanten Applikationen. So muß man sich für eine Reihenfolge der Detaillierungsdimensionen entscheiden, z.B. zuerst produktmäßig, dann regional und zuletzt zeitmäßig. Dabei geht jedoch der Faktor "datengetrieben" bezüglich der Anzahl verfügbarer Dimensionen verloren.

Eine konsequente Weiterentwicklung mit zusätzlichen Vorteilen für die Wartungseffizienz läßt sich dadurch erreichen, daß diese Dimensionsinformation, bzw. in verallgemeinerter Form Drill-Down-Information, attributmäßig je Informationseinheit in der EIS-Datenbasis hinterlegt wird. Der Drill-Down könnte dann als eigenständiges Konstrukt mit standardisierter Anwendungsoberfläche, z.B. als Popup-Window, realisiert werden, das exakt auf diese Attributinformation der selektierten Informationseinheit zugreift und die aktuell verfügbaren Drill-Down-Möglichkeiten zur Auswahl anbietet bzw. direkt auslöst. Dadurch lassen sich insbesondere heterogene Drill-Down-Verzweigungen aus homogenen Informationsdarstellungen realisieren, d.h. der Umsatz eines Produktes mit regionalen Ausreißern könnte (primär) eine Detaillierung bezüglich der Regionsdimension, der Umsatz eines Produkts mit zeitlichen Schwankungen (zusätzlich) eine Detaillierung bezüglich der Zeitdimension offerieren. Weiterhin ist es denkbar, die Attributeinträge durch die Verfügbarkeit entsprechender Informationen zu steuern, z.B. im Falle der Ergänzung durch einen textuellen Kommentar, zusätzlich diesen (automatisch) anzubieten.

Werden die so entstehenden, vom EIS-Anwender potentiell beschreitbaren Verknüpfungen der abrufbaren Bildschirmeinheiten aufgezeichnet, entsteht zunächst eine Baumstruktur. Zum Prinzip von EIS-Generatoren gehört es, daß der individuell vom Benutzer beschrittene Weg durch diesen Baum kontinuierlich protokolliert wird und jederzeit angezeigt werden kann. Dieses als PATH- oder RETRACE-PATH bezeichnete Konstrukt bietet als vorgefertigte Dialogfunktion dann jederzeit die Möglichkeit, an jede beliebige Stelle des beschrittenen Pfades zurückzuspringen. Die Abb. 5-10 und 5-11 zeigen Realisierungen des Retrace-Path-Konstrukts am Beispiel der EIS-Generatoren CommanderEIS und LightShip. In LightShip sind auf den ersten Blick die individuellen "Instanzen" parametrisierter Bildschirme mitprotokolliert, während in CommanderEIS nur die jeweils letzte Einstellung verfügbar ist. Es handelt sich jedoch in LightShip nicht um das originäre Konstrukt, sondern um programmierte MENU-Actions.

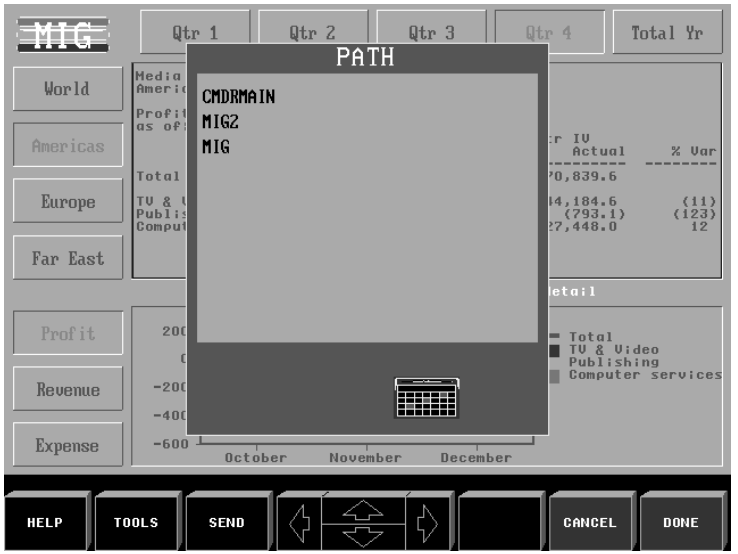


Abb. 5-10: Beispiel des Retrace-Path-Konstrukts in CommanderEIS

Da EIS-Anwendungen Drill-Down und sonstige Bildschirmwechsel, z.B. zum Wechsel von Produkt oder Region kombiniert anbieten, verwandelt sich die baumartige Drill-Down-Struktur zum einen in ein Netzwerk, zum anderen entstehen unerwünschte Redundanzen in dem aufgezeichneten Pfad. So würde dieser durch wiederholten, zyklischen Produktwechsel Schleifen protokollieren. Um dies zu verhindern bieten alle EIS-Generatoren als integrale Komponente des Retrace-Path-Konstrukts dem EIS-Entwickler bei der Sprungspezifikation die Möglichkeit, alternativ zum standardmäßigen Anhängen an den Pfad, das letzte Element oder den gesamten Pfad zu ersetzen.

Im Falle des Wechsels in einen anderen Ast der Drill-Down-Baumstruktur auf detaillierter Ebene, z.B. von Ist- zu Planwerten, ist es wünschenswert, nicht nur den bisherigen Pfad der Istwerte an dessen Ende durch den Planwert-Bildschirm zu ersetzen, sondern ganze Pfadsequenzen, im Beispiel bis hinauf zur Ist-Plan-Verzweigung, gegen Planwerte auszutauschen, d.h. praktisch ein Drill-Up zu ermöglichen. Dies ist bisher in keinem angebotenen EIS-Generator möglich. Zur Realisierung bietet sich das o.g. Drill-Down-Konstrukt an, indem zusätzlich zu dem Nachfolger-Attribut ein Vorgänger-Attribut eingeführt wird. Das resultierende

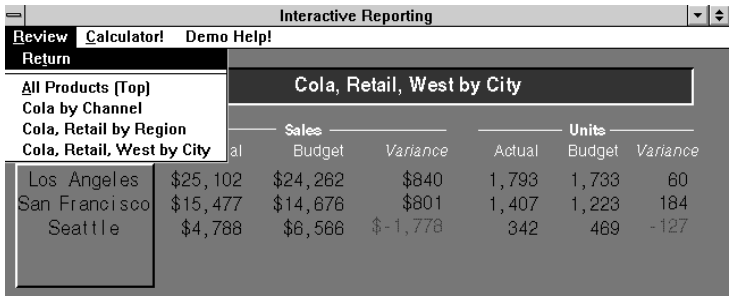


Abb. 5-11: Beispiel des Retrace-Path-Konstrukts in LightShip

kombinierte Drill-Down-/Up- oder besser Navigations-Konstrukt stellte dann eine kontextbezogene Alternative zu dem herkömmlichen, chronologie-fixierten Retrace-Path-Konstrukt dar.

Ein Spezialfall dieses vorgeschlagenen allgemeinen Navigations-Konstrukts ist mit der ExecuView-Komponente des EIS-Generators CommanderEIS gegeben. Sie basiert auf mehrdimensionalen Modellen der Planungssprache OneUp als EIS-Datenbasis. Die Vorgänger-/Nachfolger-Attribute sind implizit in den hierarchischen Verknüpfungen der Dimensionsausprägungen des Modells enthalten und werden direkt als alternierender Drill-Up/Down-Button je Dimension in ein fest vorgegebenes Oberflächenlayout umgesetzt (vgl. Abb. 5-12). Der Spezialfall ist dadurch gegeben, daß zum einen ausschließlich numerische Informationen auf diese Weise organisiert werden können, zum anderen müssen alle Daten gleich dimensioniert sein. Diese Einschränkung entfällt bei dem Produkt macControl, bei dem darüber hinaus das Layout dieses Anwendungstyps individuell gestaltet werden kann; dafür ist aber die gesamte Funktionalität nicht als Modul verfügbar, sondern muß im Rahmen der Makroprogrammierung vom EIS-Entwickler selbst erstellt werden.

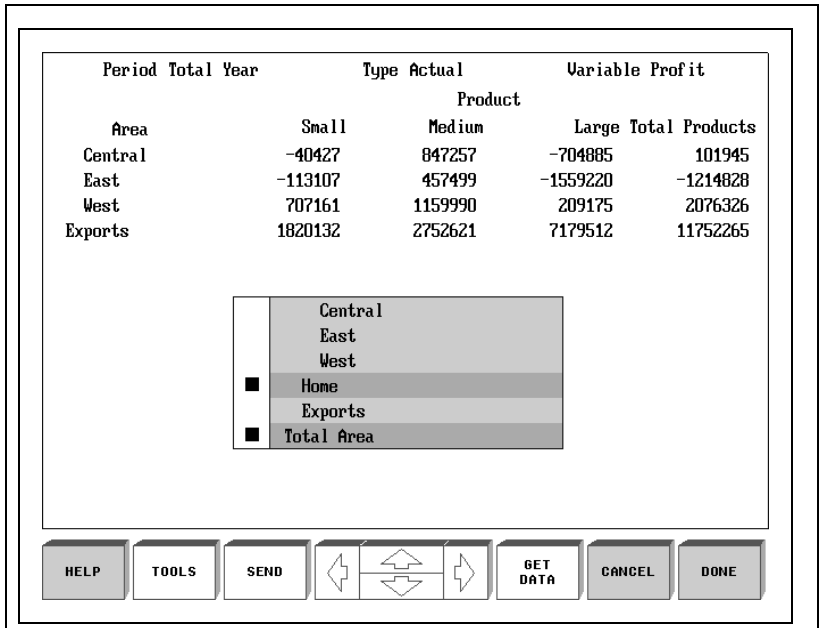


Abb. 5-12: Modellhierarchie-getriebenes Drill-Down in der ExecuView-Komponente von CommanderEIS

5.3.5. Exception-Reporting

Das wichtigste Prinzip der Informationsselektion orientiert sich an Ausnahmetatbeständen. Hierunter werden im allgemeinen *"Abweichungen von üblichen, erwarteten (prognostizierten) oder vorgegebenen (geplanten) Ergebnissen"* verstanden ([Mertens 91, Planungs- und Kontrollsysteme], S. 60). Der Abweichungsbegriff ist zumeist auf elementare quantitative Objekte, z.B. Planabweichung des Ist-Umsatz, konzentriert. In Führungsinformationssystemen treten zusammengesetzte, quantitative Objekte, z.B. Planabweichung der kumulierten Ist-Umsatzrendite bzw. aus diesen abgeleitete, qualitative Objekte, z.B. Abweichung von der angestrebten "Wettbewerbsposition" in den Vordergrund. Da der Zweck von Ausnahmeorientierten Führungsinformationssystemen darin besteht, unternehmerisches Handeln auf wesentliche Bereiche zu lenken und anschließende Aktionen zu unterstützen bzw. auszulösen, müssen sie neben der Darstellung von kritischen Abweichungen auch unmittelbar Aktionen induzieren können, z.B. automatisch das Informationsangebot kontextbezogen für eine Detailanalyse erweitern oder Themenliste und Teilnehmer einer Besprechung vorschla-

gen. Unter der Komponente Exception-Reporting eines Führungsinformationssystems soll deshalb im folgenden das gesamte Spektrum von Konstrukten verstanden werden, durch das der Anwender selbst beliebig komplexe Ausnahmebedingungen definieren und im Sinne einer ereignisgesteuerten Oberfläche mit beliebigen Aktionen verbinden kann.

Um eine differenzierte, klassifizierende Bewertung der Exception-Reporting-Komponenten angebotener EIS-Generatoren zu ermöglichen, sollen folgende Kriterien näher betrachtet werden:

- *Anzeigeform*
- *Ausnahmeobjekt*
- *Margen-Spektrum*
- *Margen-Hierarchie/-Variabilität und*
- *Funktionalität.*

5.3.5.1. Anzeigeform

Die meisten EIS-Generatoren beschränken sich beim Hinweis auf Abweichungen von individuell gesetzten Grenzwerten auf eine farbliche Markierung, entweder des Datums selbst oder durch vorgesetzte Flags. Analytische Zusatzinformationen, z.B. in Form von Legenden zur Farbinterpretation bezüglich Richtung und Intensität der Abweichung müssen bereits häufig durch den EIS-Entwickler selbst ergänzt werden. Darüberhinaus wünschenswert erscheint die Anzeige der jeweiligen Grenzwerte, vorliegender Trends oder von Hinweisen auf die Existenz von sich kompensierenden Abweichungen auf tieferen Detaillierungsstufen. Unter Berücksichtigung der Ausführungen in Kap. 5.3.4 zum Drill-Down impliziert dies, daß auch relevante Dimensionen für ein zielgerichtetes Drill-Down visualisiert werden: Ein Beispiel hierfür stellen im Sinne der Produkt-Grenzwerte tolerable Abweichungen dar, die sich in einzelnen Regionen zu Ausnahmen summieren, dann bei der Regionsaggregation jedoch wieder ausgleichen, oder umgekehrt. Aufgrund der Vielzahl derartiger Zusatzinformationen bietet es sich an, diese lediglich abrufbar im Hintergrund zu halten und den Anwender über eine einfache oder kombinierte Markierung darauf hinzuweisen.

Abgesehen von farblichen Markierungen unmittelbar angezeigter Bildschirmobjekte müssen alle genannten weiteren Fälle bislang weitesgehend durch den EIS-Entwickler selbst im Einzelfall programmiert werden. Ein Ansatz zur konstrukt-mäßigen Realisierung, d.h. automatischen Generierung von visuellen Hinweisen auf Ausnahmesituationen auf tiefer liegenden Drill-Down-Ebenen ist im EIS-Generator Forest&Trees zu finden (vgl. Abb. 5-13). Dabei wird durch eine standardisierte Ikone jedoch lediglich auf deren Existenz hingewiesen. Lokalität, Richtung und Ausmaß der tieferliegenden Abweichung können nicht nach oben

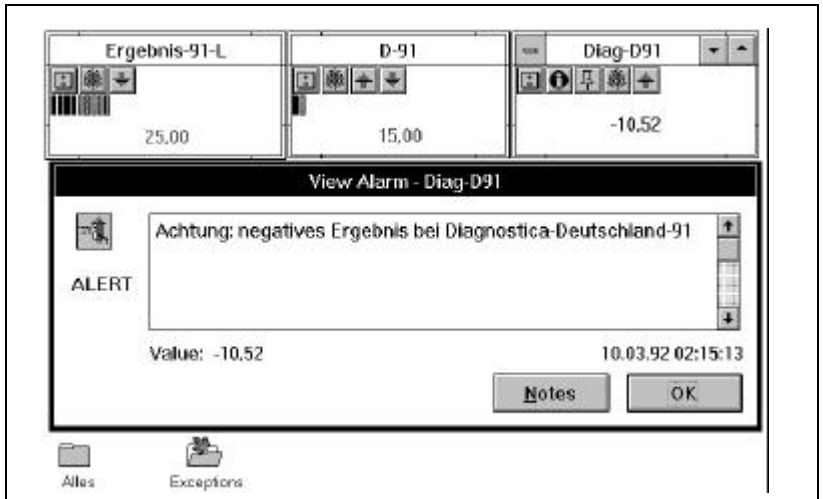


Abb. 5-13: Exception-Reporting-Konstrukt des EIS-Generators Forest&Trees

durchgereicht werden. Dafür können unmittelbar angezeigte Ausnahmeobjekte mit einer textuellen, per Ikone abrufbaren oder bei Systemstart automatisch angezeigten Zusatzinformation versehen werden, die ihrerseits automatisch die zugrundeliegende Ausnahmebedingung enthält.

5.3.5.2. Ausnahme-Objekt

Mit Ausnahme-Objekt soll das vorgesehene Spektrum von Datenarten bezeichnet werden, für das Ausnahmebedingungen festgelegt werden können. Das Spektrum reicht prinzipiell von Absolutwerten mit und ohne Dimension (Wert- und Mengeninformationen) über absolute oder prozentuale Soll-Ist-Vergleiche bis hin zu absoluten oder prozentualen Änderungsraten. Die explizite Unterscheidung von Absolutwerten und (auch absoluten) Differenzwerten erscheint aus der Praxis heraus gerechtfertigt, in der neben Abweichungen häufig großer Wert auf deren Basis gelegt wird, häufig sogar ausschließlich die Beobachtung dieser Basis präferiert wird. Beispiele hierfür sind aus Kapazitätsauslastungsgründen angestrebte Mindestmengenziele, die anstelle der abstrakteren, aggregierten Kennzahl des Kapazitätsauslastungsgrads ohnehin notwendige Analyseschritte vorwegnehmen und einen direkteren Bezug zu notwendigen Aktionen aufweisen, auch wenn dadurch implizit vorab eine Festlegung auf eine bestimmte Handlungsalternative festgelegt wird.

Mit Ausnahme von Muster-Applikationen im Produkt Pilot findet sich in den angebotenen EIS-Generatoren bisher keine typmäßige Differenzierung der Exception-Konstrukte im Sinne von Ausnahmeobjekt-Klassen. Somit müssen typmäßig identische Besonderheiten in jedem Einzelfall vom EIS-Entwickler aufwendig selbst programmiert werden. Die Musterapplikation ADVANTAGE von Pilot sieht Grenzwertpaare für absolute und prozentuale Abweichungen vor, die klassenmäßig unterschiedliche Zusatzauswertungen induzieren. Allerdings gelten die Margen uniform für alle Datensätze auf allen Hierarchie-Ebenen der Datenbasis (vgl. Abb. 5-14).

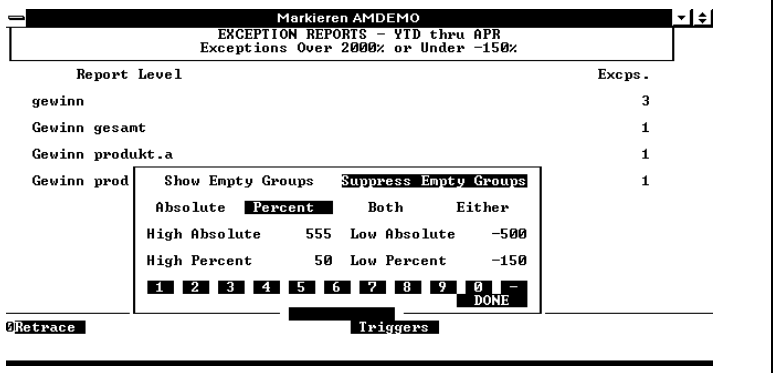


Abb. 5-14:	Exception-Reporting-Konstrukt im EIS-Generator Pilot

Ein potentielles Element für einen effizienten Konstrukttyp ist somit bspw. in der Möglichkeit zur Differenzierung von Grenzwerten bei prozentualen Abweichungsobjekten bezüglich der Basisobjekte zu sehen: so ist etwa eine 5%-ige Planabweichung unterschiedlich zu beurteilen, je nachdem ob sie sich auf ein umsatzstarkes oder ein umsatzschwaches Produkt bezieht bzw. ob es sich um ein einzelnes Produkt oder um eine hochaggregierte Produktgruppe handelt. Diese Aspekte werden durch die nachfolgend beschriebenen Kriterien des Margen-Spektrums und der Margen-Hierarchie angesprochen.

Aus den unterschiedlichen Größenordnungen von Absolutwerten resultieren ebenfalls erhöhte Anforderungen an die Funktionalität des Exception-Reporting. Im Gegensatz zu Differenzwerten können seltener und weniger Ausnahmeobjekte mit den gleichen Grenzwerten

versehen werden. Dieser auf den ersten Blick konstruiert erscheinende Aspekt leitet unmittelbar über zu dem Kriterium des Margen-Spektrums.

5.3.5.3. Margen-Spektrum

Unter Margen-Spektrum oder -Reichweite soll die Anzahl und Art der Objekte (Ausnahme-Objekte und -Klassen) verstanden werden, für die ein individuelles Grenzwertepaar (Margen) spezifiziert werden kann. Das eine Extrem wird von EIS-Generatoren bzw. -Applikationen gebildet, die dem Anwender die Spezifikation nur eines Grenzwertpaares je Ausnahmeobjektklasse erlauben, z. B. ± 5% bei allen Soll-Ist-Abweichungen, egal ob es sich um Produkte mit großem oder kleinem Umsatzanteil handelt. Das andere Extrem verlangt Grenzwerte für jedes einzelne Ausnahmeobjekt. Dazwischen liegen Werkzeuge, bei denen Margen für Sequenzen oder beliebig zusammenstellbare Gruppen von Ausnahmeobjekten spezifiziert werden können. Letzteres ist beispielsweise bei den Produkten CommanderEIS und LightShip vorgesehen (vgl. Abb. 5-15). Allerdings kann der Gruppenumfang jeweils nur durch den EIS-Entwickler festgelegt und modifiziert werden, der EIS-Anwender ist auf die Änderung der Grenzwerte selbst oder wie im Falle von CommanderEIS auf die Spezifikation von Margen für singuläre Objekte beschränkt.

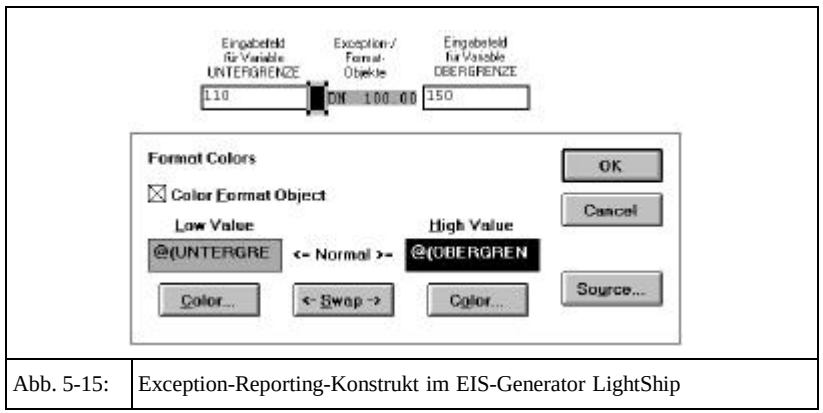


Abb. 5-15:	Exception-Reporting-Konstrukt im EIS-Generator LightShip

5.3.5.4. Margen-Hierarchie

Margen-Hierarchie zielt darauf, wer die Grenzwerte festlegt bzw. festlegen kann. Angesichts der Informationsmenge macht es wenig Sinn, dem EIS-Anwender allein die Last zur Spezi-

fikation der Grenzwerte aufzubürden; andererseits darf ihm auch nicht die Möglichkeit persönlicher Margen genommen werden. Als Lösung bieten sich zwei Konzepte an, die optimalerweise kombiniert zum Einsatz kommen sollten:

❒ zum einen die bereits angesprochene Bildung von Ausnahmeobjekt-Klassen,

❒ zum zweiten die Einführung von Margen-Versionen.

Dieses zweite Konzept sieht alternative, evtl. hierarchisch organisierte Sätze (Sets) von Grenzwerten vor: globale Vorfilter, die vorzugsweise von den Informationslieferanten im Sinne und Interesse herkömmlicher Datenverantwortlichkeiten festgelegt werden, sowie lokale, individuellen Margen des EIS-Anwenders. Ein Beispiel für den zweiten Fall ist durch das Produkt CommanderEIS gegeben, bei dem der Anwender jederzeit zwischen einem globalen (DOCUMENT) Margensatz, der vom EIS-Entwickler definiert wird, und einem lokalen (PERSONAL) Satz von Grenzwerten wählen kann, die er selbst - wenn auch nur auf Einzelobjektbasis - erzeugen und modifizieren kann (vgl. Abb. 5-16).

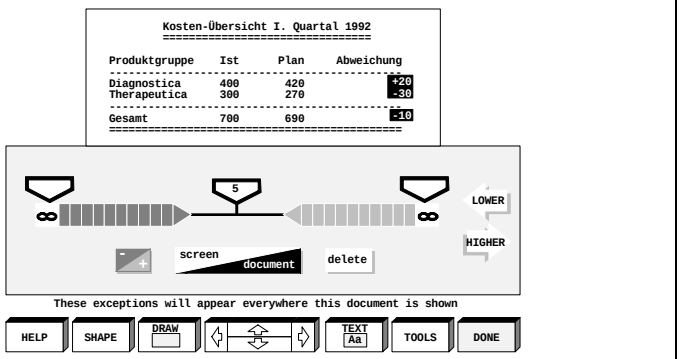


Abb. 5-16: Exception-Reporting-Konstrukt im EIS-Generator CommanderEIS

5.3.5.5. Funktionalität

Das unter Nutzenaspekten wichtigste Kriterium stellt die Funktionalität des Exception Reporting dar. Es können vier (kombinierbare) Ausbaustufen unterschieden werden:

❒ *markierendes Exception-Reporting*
Reine Visualisierung der Abweichungen am jeweiligen Punkt innerhalb des Berichtswesens. Beispiele sind die Produkte CommanderEIS und LightShip.

❒ *informatives Exception-Reporting*
Ereignisorientierte Generierung von Hinweisen über vorliegende Abweichungen. Ein Beispiel hierfür ist das Produkt Forest&Trees (vgl. Abb. 5-13).

❒ *selektives Exception-Reporting*
Automatische Selektion aller Abweichungen und deren übersichtsartige Zusammenstellung und Präsentation in Form eines Auswahlmenüs, von dem aus direkt an den Punkt der Abweichung im Berichtswesen gesprungen werden kann. Dies setzt in der Regel eine Datenbank-orientierte Datenbasis voraus. Beispiele hierfür sind die Musterapplikationen des Produkts Pilot.

❒ *erklärendes Exception-Reporting*
Abrufmöglichkeit von Informationseinheiten, die das Zustandekommen der Abweichung kausal erklären können. Sie setzen in der Regel eine Modell-Datenbasis voraus. Beispiele sind die Produkte ExecutiveEdge bzw. Paradigm.

5.4. Informations-Präsentation

5.4.1. Zielgruppen-spezifische Anforderungen

Der Anwenderkreis von Führungsinformationssystemen zeichnet sich mehrheitlich durch geringe, praktische DV-Kenntnisse bzw. -Erfahrungen und nur gelegentliche Nutzung von DV-Geräten aus (vgl. [Bullinger 91, Chefinformationssysteme], S. 19). Daran wird sich angesichts der Verteilung der Aufgaben- und Arbeitsschwerpunkte auch durch moderne Kommunikations- und Informationsmedien wenig ändern (vgl. Kapitel 2). Insofern kommt einer leicht bedienbaren und verständlichen Gestaltung der Benutzungsoberfläche unter Akzeptanzgesichtspunkten eine hohe Bedeutung zu. EIS-Werkzeuge tragen dem Rechnung durch kommandofreie, graphische Oberflächen mit standardisierten, bildhaften Symbolen zur Navigation durch und Steuerung von EIS-Anwendungen.

Andere Untersuchungen weisen gar eine Tastaturphobie nach (vgl. [Müller-Böling 90, IuK-Anforderungen für Führungskräfte], S. 97ff.). Ob die derzeit angebotenen Alternativen in Form von berührungs-empfindlichen Touch-Screens oder Infrarot-Fernbedienungen (Touchpads) diesbezüglich eine wirkliche Alternative darstellen, bleibt zu bezweifeln. Ihr Nutzen ist eher als Bedienungselement für Vorführungen und Präsentationen im Rahmen

von Konferenzen und Besprechungen mit Großbildprojektion zu sehen. Eine echte Alternative ist erst durch Notepads, d.h. Geräte mit Klarschrifteingabe zu erwarten.

Eine höhere Bedeutung ist den Gestaltungs- und Bedienungsmöglichkeiten der EIS-Oberfläche zuzumessen. Werkzeuge können zunächst nach der kleinsten gestaltbaren Bildschirmeinheit unterschieden werden: die meisten - auch solche, die unter der graphischen, Fensterorientierten Oberfläche MS-Windows operieren - orientieren sich noch an ganzen Bildschirmen und "modalen" Dialogboxen, d.h. einer strengen Benutzerführung, bei der immer nur eine Dialogaktion gleichzeitig zur Verfügung steht. Lediglich das Produkt Forest&Trees stellt hier eine Ausnahme dar, indem jedem Informationselement ein eigenes Bildschirmfenster zugeordnet wird, so daß der Anwender Auswahl, Anordnung und Größe der betrachteten Informationen individuell bestimmen kann. Im Rahmen Bildschirm-orientierter Systeme wird dieses "Zurechtschneiden" durch individuell spezifizierbare Auswahlmenüs auf Basis ganzer Bildschirme, z.B. in Form der Musterapplikation MARK/JUMP im Produkt Pilot, oder auf Basis individuell spezifizierbarer Teilsichten, z.B. bei der Modell-basierten Komponente ExecuView im Produkt CommanderEIS, unterstützt. Hierauf wird in Kapitel 5.5.1 näher eingegangen.

5.4.2. Präsentations-Konstrukte

Zur Gestaltung der Informations-Präsentation stellen alle Werkzeuge dem EIS-Entwickler einen Satz parametrisierter Objektklassen mit vordefiniertem Funktionsumfang zur Auswahl. Der Entwickler kann dann aus beliebigen Kombinationen von Instanzen dieser Objektklassen jeweils einzelne Bildschirme konfigurieren und diese über Dialogkonstrukte (s. nächstes Kapitel) miteinander verknüpfen. Unterklassen im Sinne objektorientierter Programmierung können nicht spezifiziert werden, so daß strukturell ähnliche Präsentationen bzw. Präsentationsteile durch Kopien einzelner Bildschirmspezifikationen zu wartungsintensiven Redundanzen führen. Das Produkt LightShip vermindert diesen Schwachpunkt durch die Möglichkeit der Parametrisierung, allerdings nur auf der Ebene der Bildschirm- bzw. Fenster-Spezifikation; die elementaren Präsentationsobjekte selbst können auch hier nicht subklassifiziert werden. Im einzelnen lassen sich folgende Präsentationsobjekte incl. Funktionen typischer EIS-Werkzeuge unterscheiden:

- *DOCUMENT (LightShip, CommanderEIS)*
  Sie dienen der Darstellung importierter Texte, Daten und Graphiken der EIS-Datenbasis. Für numerische Informationen bieten sie zusätzlich die Möglichkeit zur Spezifikation eines markierenden Exception-Reporting. Bei rein Dokument-orientierten Systemen, wie z.B. CommanderEIS, muß die Formatierung mit Ausnahme der Schriftgröße

und -farbe bereits vollständig im importierten Dokument realisiert sein. Bei Produkten wie LightShip, die ihre Daten neben Dokumenten auch über eine Abfragespezifikation, z.B. via DDE oder LENS (vgl. Kapitel 5.3.2), beziehen, ist eine (nachträgliche) Formatierung im EIS möglich. Hierzu steht ein FORMAT-Objekt zur Verfügung, über das neben Schriftart, - größe und -farbe auch Spaltenbreite, Datenformat sowie Präfixe und Suffixe lokal festgelegt werden können. Dabei kann sich ein FORMAT-Objekt auf beliebige Teilmengen auch mehrerer DOCUMENT-Instanzen erstrecken. Das Produkt Forest&Trees sieht hingegen Datentypen mit festgelegten Formatierungen vor, z.B. DECIMAL, INTEGER, PERCENT oder Währungsobjekte.

- *GRAPH*
  Sie dienen der Erzeugung lokaler Geschäftsgraphiken auf der Basis importierter Daten. Bei Dokument-Basen muß sich dieses ebenfalls - evtl. verdeckt - auf dem EIS-Bildschirm befinden, bei Datenbank- oder Abfrage-Basen kann die Graphik selbst auf die Datenquelle zugreifen. Graphiktypen und Gestaltungsmöglichkeiten sind in allen Fällen sehr beschränkt, so daß häufig auf den Import von mit externen Programmen erstellten Graphiken in Form von DOCUMENT zurückgegriffen werden muß.

  Da einerseits die Auswahl eines geeigneten Graphiktyps für Einheitlichkeit und Verständlichkeit der EIS-Applikation von eminenter Bedeutung ist, andererseits EIS-Entwickler sich zunehmend aus Fachabteilungen rekrutieren und nicht über entsprechende Schulung verfügen, bieten sich zwei Erweiterungen der betroffenen Konstrukte von EIS-Werkzeugen an: zum einen empfehlende Hinweise über aufgrund der Datentypen geeignete Graphiktypen (vgl. [Schorr 88, Graphiktypen]); zum anderen kann diese Information bereits als Attribut dem Informationsobjekt selbst mitgegeben werden, so daß die Graphikgenerierung quasi "objektgetrieben" vonstatten gehen kann (vgl. Kap. 9).

- *IMAGE (LightShip)*
  Sie dienen dem Import von Bildern üblicher Graphikformate, z.B. Ikonen für Unternehmenslogos oder EIS-Funktionen sowie Produktanimationen und Symbole für Ausnahmestati oder eingescannte Informationen und erlauben in der Regel Reformatierungen bezüglich der Größe.

5.4.3. Dialog-Konstrukte

Bei den Dialog-Konstrukten ist zu unterscheiden in unveränderlich und standardmäßig vorgegebene und individuell vom EIS-Entwickler gestalt- und einsetzbare. Erstere betreffen hauptsächlich die Aktivierung von vorgefertigten Verarbeitungs-Funktionen, auf die im nächsten Kapitel 5.5 eingegangen wird. Für letztere ist ein breites, allgemein anerkanntes

Spektrum standardisierter Alternativen entwickelt worden (vgl. [Hartson 90, Benutzeroberfläche]). Bezüglich EIS-Werkzeugen läßt sich derzeit folgende Klassifizierung feststellen:

❒ *HOTSPOT (CommanderEIS, LightShip)*

Hotspots realisieren in der Regel maus-sensitive Felder, über die der EIS-Anwender Aktionen auslösen kann. Bei CommanderEIS werden sie durch Zuordnung von Scripts, die die Aktionen beschreiben, zu Bildschirmobjekten realisiert. Dabei wird im Falle der Verknüpfung mit einem anderen Dokument (PATH...) über dessen Dokumenttyp bestimmt, ob ein anderer Bildschirm angezeigt, die Schnittstelle ExecuView zu einem Planungssprachenmodell oder eine Commander- bzw. DOS-Kommandosequenz gestartet werden soll (vgl. Abb. 5-17). Hotspots verfügen hier über keinerlei automatische oder klassenmäßige Funktionalität zur Visualisierung.

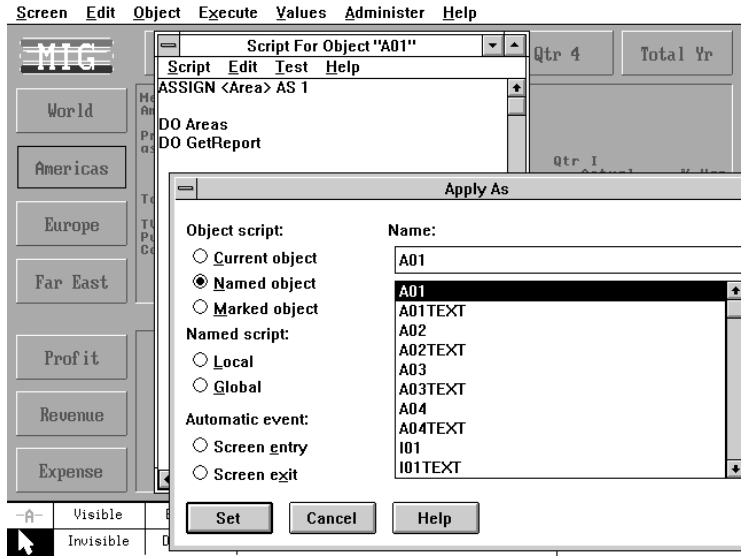


Abb. 5-17: Hotspot-Definition in CommanderEIS

Diesem im Interesse der leichten Bedienbarkeit, Entwicklereffizienz sowie Anwendungs-Standardisierung wichtigen Element trägt bspw. das Produkt LightShip durch Hotspottypen, z.B. in Form von Buttons, Checkboxen sowie Radio- und One-Shot-

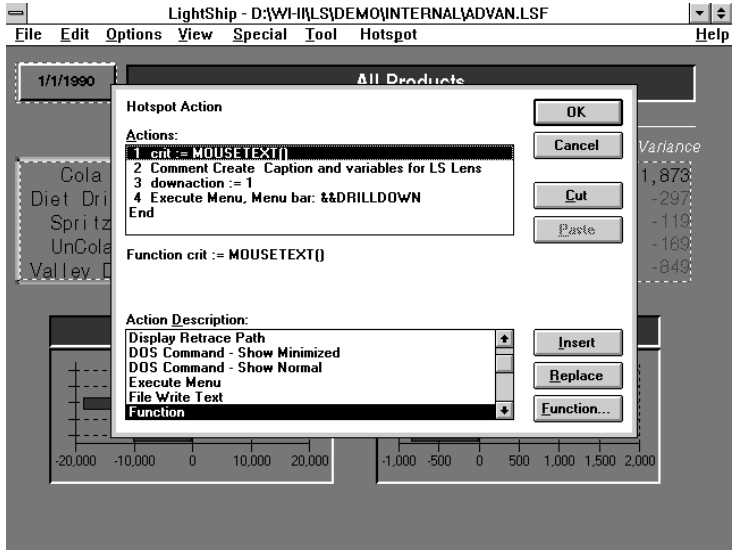


Abb. 5-18: Hotspot-Definition in LightShip

Buttons Rechnung, die durch auch Entwicklerseitig spezifizierbare Cursortypen visualisiert werden.

Bei LightShip schließt das Aktionsspektrum insbesondere die Spezifikation und Verarbeitung von Dialogparametern in Form von kleinen Unterprogrammen, z.B. für Verzweigungen, sowie den gesamten Funktionsumfang von DDE ein (vgl. Abb. 5-18).

❒ *MENUS (LightShip)*

Sie decken den gleichen Funktionsumfang wie Hotspots ab und unterscheiden sich von diesen lediglich durch ihre Anordnung in der Kopfzeile der EIS-Anwendung. Bei LightShip ergänzen sie den standardmäßig vorgegebenen Funktionsumfang.

❒ *SCROLLBARS (CommanderEIS, LightShip)*

Sie erlauben das horizontale und vertikale Positionieren von Bildschirmfenstern über importierten Daten. Bei CommanderEIS ist diese Funktion zentral als Standardfunktion realisiert, so daß zusätzlich ein Markieren des zu rollenden Objekts notwendig wird. LightShip gestattet lokale Scrollbars je Objekt; außerdem kann der rollbare Dokumentausschnitt vom Entwickler spezifiziert werden.

Im Falle von Dokumenten als Datenbasis enthalten diese in der Regel Beschreibungen in Kopfzeilen und Vorspalten, die nicht mitgerollt werden dürfen. Um den Mehrfachimport in rollbare und nichtrollbare Fensterausschnitte, wie bspw. bei LightShip notwendig, zu vermeiden, verfügt CommanderEIS hierzu über Parameter zur Dokumentinterpretation, die den Rollbereich festlegen (vgl. Abb. 5-19). Dieser ist jedoch nicht dynamisch, so daß Änderungen der Dokumentstruktur wieder Wartungsaufwand nach sich ziehen.

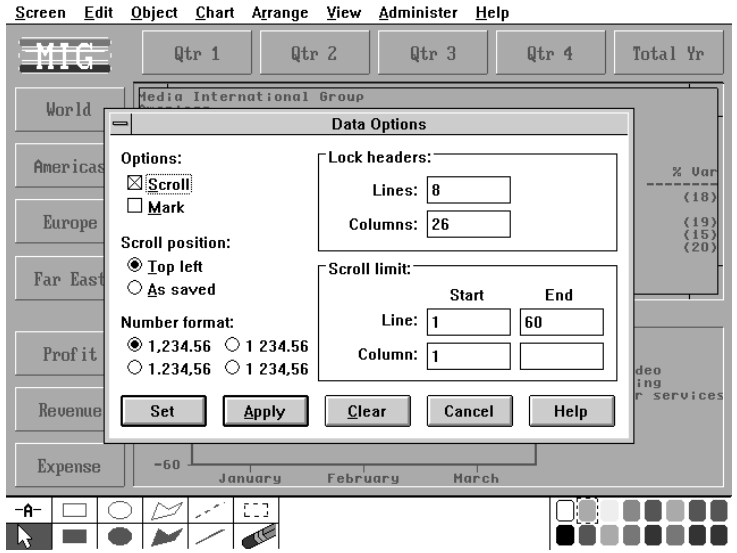


Abb. 5-19: Dokument-Import-Parameter in CommanderEIS

❒ *Meldungen, Fehler, Hilfe*

Dieses in der Theorie der Dialoggestaltung unverzichtbare Element fehlt erstaunlicherweise in nahezu allen EIS-Werkzeugen als eigenes Konstrukt, so daß es in Form von Popup-Windows vom EIS-Entwickler individuell und aufwendig gestaltet werden muß.

❒ *Text-/Daten-Eingabe*

Auch hier ist das Angebot sehr bescheiden und beschränkt sich auf einzeilige TEXT-ENTRY-Felder, z.B. beim Produkt LightShip, über die praktisch nur einwertige Parameter, z.B. im Rahmen des Exception-Reporting, abgewickelt werden können.

Popup-Menüs zur Parameter-Auswahl, z.B. alternativer Regionen, müssen in LightShip Entwickler-seitig aus mehreren Basisobjekten (Dokument, Hotspot,...) in jedem Einzelfall neu generiert werden. Bei Forest&Trees und CommanderEIS stehen zumindest entsprechende Funktionen bzw. SCRIPT-Anweisungen zur Verfügung. Im Falle von Forest&Trees stehen im Zuge der Einführung von View-Typen mittlerweile List-Views zur Verfügung, im Falle von CommanderEIS ist zusätzlich ein Datenbasis-Dokument vom Typ LIST notwendig, das bei Änderungen jeweils permanent aktualisiert werden muß.

❒ *Periodic Events*

Hierunter werden Einrichtungen verstanden, die es ermöglichen, zu festgelegten Zeitpunkten oder Ereignissen automatisch bestimmte EIS-Aktionen ablaufen zu lassen. Ein als Standard implementiertes Beispiel hierfür ist im Rahmen des Exception-Reporting des Produkts Forest&Trees gegeben: hierbei kann der Entwickler je Informationsobjekt festlegen, ob der Anwender beim Programmstart oder Neuberechnung automatisch über aufgetretene Abweichungen per Popup-Window informiert werden soll; dieses kann neben einem variablen, textuellen Hinweis des Entwicklers auch Notizen des Anwenders, z.B. über eigene Einschätzungen oder unternommene bzw. zu unternehmende Schritte, aufnehmen.

Eine zeitbezogene Variante ist im Produkt CommanderEIS zu finden: hierbei kann der Entwickler zeitpunkt- oder zeitraum-abhängig die Ausführung beliebiger Aktionen spezifizieren, z.B. auch den Update der lokalen Datenbasis oder die Anzeige bestimmter Bildschirme. Die Anwendung ist eng verbunden mit einer integrierten "Kalender"-Komponente, auf die im folgenden Kapitel näher eingegangen wird.

5.5. Informations-Verarbeitung

5.5.1. Präsentations-Sequenzen

Neben dem Abruf einzelner, vorgefertigter Informationseinheiten kommt der Möglichkeit, benannte Sequenzen dieser Informationsbildschirme selbst zusammenstellen und verwalten zu können, hohe praktische Bedeutung zu. Anwendungsbeispiele sind Besprechungen oder Präsentationen, bei denen EIS-Inhalte, kontextbezogen selektiert und um externe Elemente, z.B. Schriftfolien ergänzt, direkt als Argumentationshilfen dienen können. Deren unmittel-

bare Herkunft aus einem unternehmensweit, d.h. z.B. allen Beteiligten der Besprechung verfügbaren EIS kann die Konsensfähigkeit erhöhen und die Akzeptanz fördern. Die Effizienz der Vorbereitung regelmäßiger Konferenzen kann gesteigert werden, wenn die Zusammenstellung auf Spezifikationsebene geschieht, d.h. nicht konkrete Dokumente, sondern sich aufgrund des jeweiligen Datenbestandes selbst aktualisierende Abfragen zum Einsatz kommen. Bei Ausweitung der Zusammenstellungs-Spezifikation um selektive Komponenten des Exception-Reporting sind dynamisch generierte Tagesordnungen denkbar.

Ein Realisierungsbeispiel liegt mit der Musterapplikation MARK/JUMP im EIS-Werkzeug Pilot vor (vgl. Abb. 5-20). Dabei kann der EIS-Anwender jeden abrufbaren EIS-Bildschirm unter einem individuellen Namen in ein spezielles Auswahlmenü übertragen, wobei systemseitig nicht nur dessen Abfragespezifikation gespeichert wird (MARK), sondern auch alle Verzweigungsmöglichkeiten, z.B. für Drill-Down erhalten bleiben. Innerhalb dieses Menüs (JUMP) können die Einträge umbenannt, sortiert und farblich markiert werden. Allerdings ist nur eine Menüsequenz je Anwender möglich. Ein generisches Konstrukt für beliebig viele, benannte Präsentationssequenzen wurde auf Basis des Werkzeugs macControl implementiert (vgl. [Pauliks 91, Marketing-Controlling], S. 42ff.).

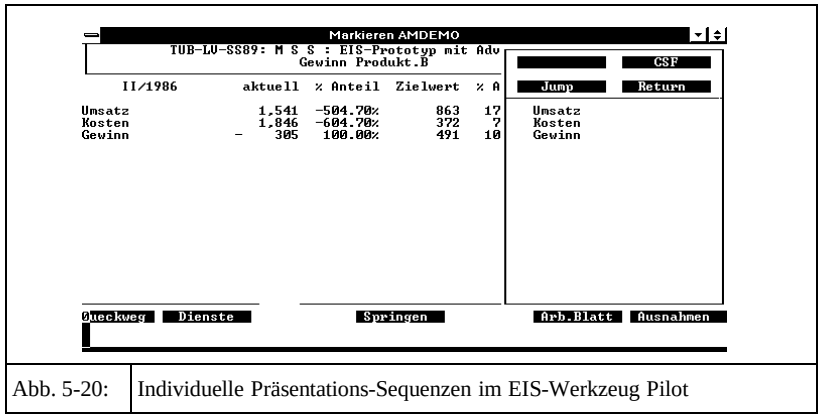


Abb. 5-20:	Individuelle Präsentations-Sequenzen im EIS-Werkzeug Pilot

Ein alternatives Beispiel ist im Produkt CommanderEIS in Verbindung mit der Standardapplikation eines persönlichen Kalenders realisiert, auf das im nächsten Kapitel eingegangen wird.

5.5.2. Persönlicher Kalender

Zu den typischen Aufgaben von Führungskräften gehört es, Informationen in Bezug auf bestimmte Zeitpunkte oder Ereignisse, z.B. Budgetbesprechungen oder Planungssitzungen zusammenzustellen. Somit ist eine Kopplung der Funktionen Präsentations-Sequenz und Terminkalender gegeben. Auf diese typische Anforderung ist die Standardapplikation REMINDER des EIS-Werkzeugs CommanderEIS zugeschnitten. Dabei können einzelne EIS-Bildschirme, um kurze textuelle Notizen ergänzt, im Rahmen eines integrierten Kalenders bezüglich eines bestimmten täglichen Termins zur Wiedervorlage abgelegt werden. Die Ablage kann sich wahlweise auf die Notiz beschränken, das Bildschirmdokument inhaltlich bezüglich der Werte zum Zeitpunkt der Ablage betreffen oder dynamisch das Dokument zum Abrufzeitpunkt mit den dann aktuellen Daten versorgen.

5.5.3. Kommentierung

Elementare Anforderung im Zusammenhang mit Informationssystemen ist die Möglichkeit der Ergänzung um individuelle, textuelle Notizen oder Kommentare. Trotzdem ist eine Implementierung als generische Komponente von Informationsobjekt-Konstrukten derzeit lediglich im Produkt Forest&Trees zu finden. Hierbei kann jedem Informationsobjekt ein Kommentar hinzugefügt werden, dessen Existenz automatisch durch eine entsprechende Button-Ikone dargestellt wird. Bei CommanderEIS ist dies nur in Verbindung mit der Kalender-Funktion möglich, bei anderen Werkzeugen obliegt die Realisierung dieser Funktion weitestgehend dem EIS-Entwickler, der aufgrund der beschränkten Texteingabemöglichkeiten (s. Kapitel 5.4.3) in der Regel auf aufwendige Aufrufe entsprechender externer Textverarbeitungsprogramme zurückgreifen muß; Hauptproblem dabei ist jedoch die fehlende, automatische logische Verbindung zwischen Informationsobjekt und Kommentar.

5.5.4. Versand

Ein Großteil der Aktivitäten von Führungspersonen besteht in Kommunikation (vgl. Kapitel 2.2). Um EIS-Inhalte ohne Medienbruch in diesen Prozeß einzubeziehen, ist die Integration einer leistungsfähigen Electronic-Mail-Komponente unabdingbar. Leistungsfähig heißt dabei, daß sämtliche Informationen, also insbesondere auch Graphiken und Bilder verarbeitet werden können. Die meisten EIS-Werkzeuge beschränken sich auf die Bereitstellung von Schnittstellen zu standardisierten, marktgängigen EMail-Systemen, z.B. PROFS, ALL-IN-ONE oder MEMO. Dabei sind häufig noch Einschränkungen bezüglich des Versands von Graphik-Bestandteilen gegeben.

Das Produkt CommanderEIS verfügt aufgrund seiner integrierten, Zentralrechner-basierten Dokumentverwaltung incl. Owner-Attribut über eine eigene Versandmöglichkeit aller Dokumente. Aus Gründen der Integration in die Unternehmens-Infrastruktur, d.h. den Anschluß aller Mitarbeiter, ist jedoch grundsätzlich eine Schnittstellen-Lösung zu präferieren.

5.5.5. Adhoc-Analysen

Führungsinformationssysteme sind primär auf die Bereitstellung regelmäßig wiederkehrenden Informationsbedarfs ausgerichtet. Zwar können durch benutzerfreundliche Entwicklungsoberflächen auch adhoc auftretende Fragestellungen relativ schnell realisiert werden, diese müssen jedoch jeweils unmittelbar auf der EIS-Datenbasis aufsetzen, d.h. quasi von Null, d.h. mit der Spezifikation von Abfragen beginnen, da Sekundär-Abfragen zumeist in Verbindung mit Bildschirmspezifikationen abgelegt werden und einer Weiterverarbeitung nicht zugänglich sind. Dieser Level dürfte auch bei größter Bedienerfreundlichkeit aufgrund des begrenzten Zeitrahmens von Führungskräften auch in Zukunft ausschließlich EIS-Entwicklern vorbehalten bleiben.

Eine Ausnahme bilden die Musterapplikation WORKSHEET im EIS-Werkzeug Pilot sowie die Standard-Applikation ExecuView im Produkt CommanderEIS:

- *Die Worksheet-Musterapplikation von Pilot*
  Hierbei kann der EIS-Anwender ausgewählte Daten vom Typ Zeitreihe aus beliebigen EIS-Bildschirmen in ein Arbeitsblatt kopieren, dort durch einfache arithmetische Operationen verknüpfen und in Form einfacher Geschäftsgraphiken darstellen. Der Vorteil dieser zentralen Anwendung ist darin zu sehen, daß es möglich ist, Zeitreihendaten aus verschiedensten Bereichen des EIS zusammenzuführen, z.B. interne Produktivitätskennzahlen mit gesamtwirtschaftlichen Kennzahlen aus externen, statistischen Informationsdiensten in Beziehung zu setzen. Ferner können die so selektierten Daten für eine komplexere Aufbereitung auch in gängige PC-Spreadsheet-Pakete exportiert werden.
- *Die ExecuView-Standardapplikation von CommanderEIS*
  Im Rahmen der bereits in Kap. 5.3.2 dargestellten Standardschnittstelle zu Modellen der Planungssprache OneUp stehen dem EIS-Anwender auch Möglichkeiten der Ergänzung der Modellspezifikation zur Verfügung. Das Spektrum deckt Summen, Differenzen und Quotienten über beliebige Modelldimensionen ab. Die Resultate können in vollem Umfang von den Darstellungs- und Speicherungsmöglichkeiten von ExecuView Gebrauch machen, d.h. insbesondere in Graphiken umgesetzt und als benannte PERSONAL-Views abgespeichert werden (vgl. Abb. 5-21).

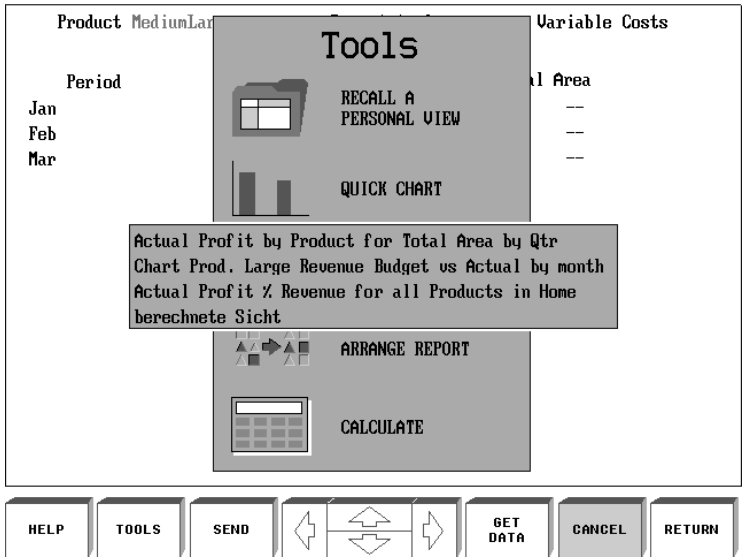


Abb. 5-21: Personal-Views in der ExecuView-Komponente von CommanderEIS

6. Generatoren für Decision Support Systems (DSS)

6.1. Referenzmodell für die Konstrukt-Analyse von DSS-Generatoren

Zur Beschreibung der charakteristischen Komponenten und Funktionen von DSS (und DSS-Generatoren) wird in der Literatur eine Vielzahl von konzeptionellen Modellen angeboten (vgl. [Rieger 85, EUS], S. 32ff. und 83ff.). Sie unterscheiden sich durch Abstraktionsgrad und Zielsetzung:

- *Prozess-orientierte Ansätze* betonen den Aspekt der Entwicklungsphasen (vgl. [Sol 82, DSS-Processes and Tools]) bzw. die adaptiven Interaktionsprozesse der beteiligten Gruppen (vgl. [Keen 80, DSS-Research-Perspective]), lassen aber strukturelle DSS-Komponenten außer Betracht.
- *Stuktur-orientierte Ansätze* stellen die Architektur-Komponenten in den Vordergrund (vgl. [Bonczek 81, DSS-Foundations]; [Borgwaldt 80, Modulare Programmsysteme]; [Sprague 80, DSS-Research-Framework]), verzichten dabei jedoch auf Aspekte der Generierung, Verwaltung und Anwendung dieser Komponenten.

Eine Analyse und Klassifizierung generischer Konstrukte in DSS-Generatoren erfordert hingegen ein Referenzmodell, das sowohl strukturelle Komponenten unabhängig von der methodischen Spezialisierung als auch funktionale Komponenten für DSS-Entwickler und -Anwender umfaßt. Für die weitere Betrachtung soll deshalb das in Abb. 6-1 dargestellte Referenzmodell Verwendung finden, das bezüglich des Abstraktionsgrads zwischen die aus der Literatur bekannten Modelle von Bonczek (vgl. [Bonczek 81, DSS-Foundations]) und Stohr (vgl. [Stohr 83, DSS-User interfaces]) einzuordnen ist. Es verfeinert die abstrakten Komponenten Knowledge- (KS), Problem-Processing- (PPS) und Language-System (LS) des Bonczek-Ansatzes in die Methoden- und Modelltyp-unabhängigen Bestandteile Modell- und Anwendungs-Basis bzw. die Entwickler-orientierten Funktionen der Modell- und Anwendungs-Spezifikation sowie die Ablauf-orientierte Interpretationskomponente Modell-Berechnung.

Die *Modellbasis* dient der Abbildung aller Elemente und Element-Beziehungen des betrachteten Unternehmens- bzw. Umweltausschnitts. Sie soll dabei sowohl Modelle im üblichen Sinne, z.B. Planungs- oder Analyse-Modelle als auch Datenmodelle umfassen. Damit wird zwei Tendenzen Rechnung getragen:

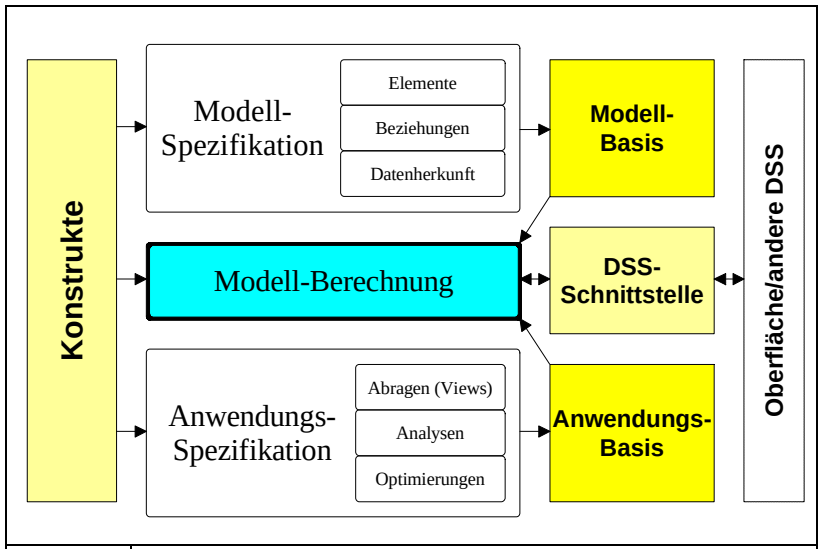


Abb. 6-1: Referenzmodell für konventionelle DSS-Generatoren

- ❒ zum einen weiten sich Datenmodelle durch die Möglichkeit der Spezifikation komplexer Berechnungen und Verknüpfungen (Relationen, semantische Netze, usw.) in und über den Bereich klassischer Modelle hinaus aus (vgl. Non-Standard-Datenbanken, z.B. [Härder 85, DBS-Architektur], S. 253ff.); dieser Prozeß wird durch virtuelle Datenbanktabellen (views) unterstützt, d.h. daß Verknüpfungsergebnisse erst zum Zeitpunkt ihres Abrufs (temporär) berechnet werden.
- ❒ zum zweiten basieren klassische Modelle im Zeichen der Datengetriebenheit zunehmend auf Struktur (Datenmodell) und Inhalt (Datensätze) der Datenbasis, d.h. Umfang und Tiefe eines Modells ergeben sich erst zur Laufzeit aus den konkreten Eintragungen in der Datenbasis.

Die *Anwendungsbasis* dient der Speicherung und Verwaltung aller Zugriffsoperationen auf die Modellbasis. Sie umfaßt sowohl standardmäßige, vom DSS-Generator bereitgestellte Funktionen (Methodenbasis), z.B. für Sensitivitätsanalysen oder Optimierungsrechnungen, als auch individuell vom DSS-Entwickler spezifizierte Funktionen, z.B. für Modellsichten (Views) oder parametrisierte Kombinationen von ergänzten bzw. mitgelieferten Funktionen. Im Vergleich zu traditionellen Referenzmodellen stellt diese (neue) Definition der Anwendungsbasis eine Kombination von Extrakten der herkömmlichen Modellbasis in Bezug auf

individuelle und der Ablaufkomponente in Bezug auf Standard-Funktionen dar. Ihre Einführung erscheint aus zwei Gründen gerechtfertigt:

- ❒ zum einen erhöht die Entkopplung von eigentlichem Modell und Modellanwendung die Mehrfachverwendung der zugrundeliegenden Modelle und senkt damit potentiell wartungsunfreundliche Redundanzen.
- ❒ zum anderen wird der Anteil individueller Anwendungsspezifikationen in DSS durch den Ausbau von Makro-, Prozedur- und Kommandosprachen bzw. deren Parametrisierung permanent erhöht.

Konkrete DSS-Generatoren realisieren ihre typische Ausrichtung bzw. Beschränkung auf bestimmte Problemklassen (vgl. [Krallmann 87, Executive Support System], S. 28ff.) durch den jeweils zur Verfügung gestellten Vorrat an vordefinierten Konstrukten für die Modell- und Anwendungs-Spezifikation bzw. Modell-Berechnung. In den folgenden Kapiteln 6.2 und 6.3 werden die wichtigsten Konstrukte von konventionellen und wissensbasierten DSS-Generatoren mit Relevanz für die Entwicklung von Führungs-Informations-Systemen nach diesen Funktionsgruppen gegliedert dargestellt. Zur Illustration dienen jeweils Beispiele repräsentativer, z.Zt. am Markt angebotener Werkzeuge.

6.2. Konventionelle DSS-Generatoren

6.2.1. Modell-Spezifikation

6.2.1.1. Mehrdimensionalität

Ein charakteristisches Merkmal von DSS-Generatoren ist deren Dimensionalität. Sie ergibt sich aus der maximalen Anzahl von Merkmalen, die zur eindeutigen Beschreibung von Modellwerten verwendet werden können. In eindimensionalen DSS-Generatoren werden alle Werte nur durch ein einziges Merkmal, üblicherweise den Variablennamen, repräsentiert, z.B. Umsatz, Kosten, Gewinn usw.. Bei mehrdimensionalen DSS-Generatoren können weitere Merkmale hinzutreten, die als (alphanumerische) Indizes der Basisdimension Variablen interpretiert werden können, z.B. Umsatz je Zeiteinheit, Produkt und/oder Region, Planversion oder Ist usw..

Unterschiede in den Dimensionalitäten von DSS-Generator und zu modellierender Problemstellung führen zu einem Abbildungsproblem, daß durch Projektion gelöst werden muß. Dabei werden zusätzliche Dimensionen in die Variablen-Bezeichnung integriert, so daß z.B. die Variablen Umsatz-Produkt1, Umsatz-Produkt2 usw. entstehen. Daraus können sich

Nachteile für den Wartungsaufwand des Modells ergeben. Die Erweiterung einer Modelldimension, z.B. um ein neues Produkt erfordert die *explizite* Spezifikation einer Vielzahl von Variablen, entsprechend der Kombinatorik zu berücksichtigender Dimensionen und deren Ausprägungen. Der gleiche Mehraufwand entsteht für die Ergänzung bzw. Anpassung strukturell identischer Modellbeziehungen, z.B. Umsatz=Absatzmenge*Verkaufspreis.

Zur Reduktion dieses Problems bieten zahlreiche konventionelle DSS-Generatoren (Planungssprachen) spezielle Konstrukte an: Im einfachsten Fall wird das Prinzip des kartesischen Produkts von Dimensionen verfolgt, d.h. jede neue Ausprägung einer zusätzlichen Dimension generiert Modellwerte für Kombinationen zu allen anderen, bestehenden Dimensionsausprägungen. Beispiele hierfür sind die Produkte CA-Compete von Computer Associates, SystemW/OneUp von Comshare oder FCS-Multi von Pilot. Während bei CA-Compete alle Dimensionen gleichwertig sind, weisen die anderen Produkte Qualifizierungen der Dimensionen in Bezug auf erlaubte Beziehungsdefinitionen auf: beliebige, arithmetische Verknüpfungen sind dort nur für die sogenannten Basisdimensionen Variable und Zeit möglich, alle weiteren Dimensionen sind auf die Addition beschränkt. Lediglich im Reportgenerator stehen zusätzlich Differenz- und Prozentbildung zur Verfügung.

Dieses kartesische Konstrukt zur Mehrdimensionalität erweist sich jedoch für solche praktische Problemstellungen ungeeignet, bei denen die Modellwerte unterschiedlich dimensioniert sind. Dabei können zwei Fälle unterschieden werden:

- ❒ *Unterschiede bezüglich der Dimensionsausprägungen*
  Ein Beispiel hierfür liegt vor, wenn nicht alle Produkte in allen Regionen vertrieben bzw. in allen Produktionsstätten gefertigt werden.
- ❒ *Unterschiede bezüglich der Dimensionen*
  Beispiel hierfür sind Umsatz je Region und Zeit vs. Fixkosten je Kostenstelle und Zeit.

Zur Lösung dieses Problems bieten o.g. DSS-Generatoren (außer CA-Compete) Konstrukte zur Einschränkung der Kombinatorik an. Allerdings umfassen sie nicht die Basisdimensionen Variable und Zeit, so daß Modellinhomogenitäten des zweiten Typs nicht unterstützt werden. Als Ausweg bleibt nur die Bildung entsprechend dimensionierter Teilmodelle, was aufgrund benötigter Verknüpfungen häufig unerwünscht ist.

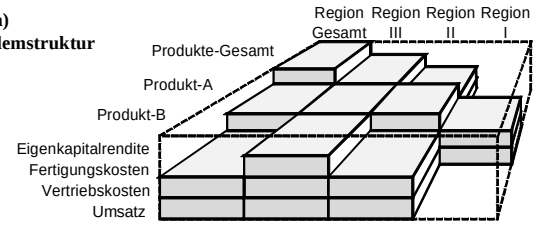
Ein flexibleres Konstrukt zur Abbildung von Mehrdimensionalität kommt im DSS-Generator macControl von Breitschwerdt&Partner zum Einsatz. Hierbei sind Variablenindizierung und Dimensions-Spezifikation nicht wie in den o.g. Werkzeugen gekoppelt, sondern stehen als getrennte Modellierungskonstrukte zur Verfügung: Jede potentielle Dimensionsausprägung wird zunächst lediglich als deskriptives "Merkmal" einer "Merkmalsgruppe" definiert. Als systemseitig vordefinierte Merkmalsgruppen stehen "Arbeitsblatt" für Modellsichten, "Bezeichnung" für eindimensionale Größen und "Zeiten" für den Zeithorizont des Modells zur Verfügung. Modellwerte samt zugehörigem physischem Speicherplatz werden durch die kombinatorische Zuordnung von Merkmalen zu einem lokalen Arbeitsblatt- oder zentralen Datenbasisfeld spezifiziert. Im Gegensatz zu den o.g. DSS-Generatoren, die auf numerische Inhalte beschränkt sind, können Datenbasisfelder beliebige Werte aufnehmen, insbesondere auch Text, Bilder und Formeln. Erst durch die Expansion bzw. Generalisierung von Arbeitsblatt- oder Datenbasisfeldern über Merkmale bzw. Merkmalsgruppen werden Modelldimensionen gebildet, und zwar prinzipiell individuell für jeden Modellwert (Variable).

Dabei sind Merkmalsgruppen nicht zwangsläufig identisch mit logischen Problemdimensionen. Vielmehr ist das Kriterium für die Bildung einer Merkmalsgruppe durch die Struktur- und Verhaltensgleichheit der zugehörigen Merkmale, d.h. gleiche oder ähnliche Beziehungsgleichungen gegeben. Diese formelmäßigen Zusammenhänge können nämlich über die Merkmalsgruppen verallgemeinert werden. Unterscheiden sich beispielsweise Produktgruppen durch die Art ihrer Berechnung oder die Menge zu berücksichtigender betriebswirtschaftlicher Größen, so kann für jede Produktgruppe eine eigene Merkmalsgruppe definiert werden, ohne die Nachteile voller Mehrdimensionalität (kartesisches Produkt) einzugehen. Diese Vorgehensweise entspricht praktisch einer Teilmodellbildung, wobei allerdings flexible Erweiterungen jedes Teilmodells sowie deren Verknüpfung durch die implizite Einbettung in ein und dieselbe Datenbasis entwicklungs- und wartungsfreundlich erhalten bleiben.

Eine Alternative zur Definition von Modelldimensionen durch explizite Benennung aller Ausprägungen im Arbeitsblatt wird durch das Konstrukt der *Zuordnungen* gegeben. Hierdurch lassen sich benannte Teilmengen einzelner oder kombinierter Merkmalsgruppen bilden, die ihrerseits über ähnliche Funktionalität wie Merkmalsgruppen verfügen. Eine spezielle Art "Zuordnung" wird unter der Bezeichnung ":Gesamt" systemseitig angeboten. Sie umfaßt immer alle Merkmale einer Merkmalsgruppe. Wird eine solche Zuordnung zur Definition eines Datenbasisfeldes verwendet, so wird dieses quasi automatisch bezüglich aller Merkmale der Zuordnung dimensioniert.

Abb. 6-2 verdeutlicht die Konsequenzen dieser alternativen Konstrukte zur Mehrdimensionalität für die Modelldatenbasen an einem konstruierten, aus Gründen der graphischen Darstellbarkeit auf drei Dimensionen beschränkten Beispiel, dessen Struktureigenschaften in der Realität von Mehrproduktunternehmen, die auf unterschiedlichen Märkten agieren jedoch eher den Regelfall darstellen:

Teil a)
Problemstruktur



Teil b) 2-D-Lösung

MEHRDIM.XLS

	A	B	C	D	E
1		Region-Gesamt	Region-III	Region-II	Region-I
2	EK-Rendite				
3					
4	Fertigungskosten-Gesamt				
5	Fertigungskosten-A				
6	Fertigungskosten-B				
7					
8	Vertriebskosten-Gesamt				
9	Vertriebskosten-A				
10	Vertriebskosten-B				
11					
12	Umsatz-Gesamt				
13	Umsatz-A				
14	Umsatz-B				
15					

Teil c) 3-D-Lösung ohne EXCLUDE

Teil d) 3-D-Lösung mit EXCLUDE

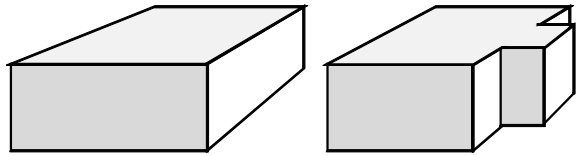


Abb. 6-2: Konsequenzen alternativer Modellierungskonstrukte in Sachen Mehrdimensionalität

Der Teil a) illustriert die abzubildende Modellstruktur: Es handelt sich um ein Unternehmen mit drei Vertriebsregionen I, II und III, wobei sich in zwei davon (II und III) auch Fertigungsstellen befinden. In einer davon werden zwei Produkte A und B gefertigt, in der anderen (II) nur das Produkt A. Dieses Produkt A wird in allen Regionen, B nur in zwei Regionen (II und III) vertrieben. Aus Analysegründen werden zusätzlich eine Region Gesamt (RG)

bzw. ein Produkt Gesamt (PG) auf Unternehmensebene eingeführt. Somit entstehen Umsätze und Vertriebskosten in allen Produkt-/Regions-Kombinationen außer B/I und PG/I, letzteres da nur ein Produkt in Region I vertrieben wird. Bei den Fertigungskosten entfallen alle Kombinationen mit der Region I, B/II sowie B/RG und PG/II. Für die Kennzahl Eigenkapitalrendite auf Gesamtunternehmensebene macht nur die Kombination PG/RG Sinn.

Teil b) der Abbildung zeigt exemplarisch eine mögliche Modellierungslösung für einen DSS-Generator mit nur zwei Dimensionen, z.B. MS-EXCEL (Spreadsheet). In ihr ist die Produktdimension in die Variablen eingearbeitet, die Regionsunterschiede resultieren in unterschiedlicher Spaltenbelegung (Balken). Alternativ können auch mehrere "rechteckige" Teilmodelle gebildet werden, allerdings mit der Konsequenz redundanter Beziehungs- und komplexer Verknüpfungs-Definitionen.

Teil c) der Abbildung verwendet einen mehrdimensionalen DSS-Generator ohne EXCLUDE-Konstrukt, z.B. CA-Compete. Hierbei muß der gesamte Würfel modell- und Speicherplatz-mäßig angelegt werden. Da Beziehungsdefinitionen in diesem Konzept grundsätzlich vektoriell sind, d.h. zunächst für alle Ausprägungskombinationen der übrigen Dimensionen gelten, müssen sie für die nicht benötigten Zellen explizit durch Einschränkungen der Beziehungsspezifikation ausgeschlossen werden. Ein Beispiel ist die Berechnung der Eigenkapitalrendite, die sonst in vielen Fällen zur Division durch Null führen würde.

Teil d) entspricht Fall c), ergänzt um das EXCLUDE-Konstrukt, z.B. OneUp. Hierbei können zwei "Säulen" des Würfels, die Kombinationen B/I und B/PG, Modell- und Speicherplatz-mäßig explizit ausgeschlossen werden. Somit reduziert sich der Aufwand für einschränkende Beziehungsspezifikationen um diese Kombinationen von Dimensionsausprägungen.

Teil a) der Abbildung schließlich entspricht der Lösung mit einem "heterogen" mehrdimensionalen DSS-Generator, z.B. macControl. Die Problemstruktur kann 1:1 modelliert werden. Explizite Einschränkungen von Beziehungsspezifikationen aufgrund heterogener Dimensionierungen sind nicht notwendig, da sie implizit durch die Variablen-individuelle Dimensionierung automatisch abgedeckt werden.

6.2.1.2. Konsolidierung

In engem Zusammenhang mit der Mehrdimensionalität steht die Konsolidierung. Dabei werden die Ausprägungen einer Dimension in einer hierarchischen Struktur angeordnet, in der jede Ebene (außer der untersten) eine Aggregation der zugeordneten Ausprägungen der

nächst-niedrigeren Ebene darstellt. Beispiele sind Produkte und Produktgruppen eines Mehrproduktunternehmens, Tochterunternehmen eines Konzerns oder Beteiligungsstrukturen einer Holding.

Grundsätzlich kann Konsolidierung als Spezialfall einer Gruppe von Verknüpfungen in mehrdimensionalen Modellen aufgefaßt werden, bei denen die Beziehungsdefinition in hohem Maße identisch für die anderen Dimensionen ist. Andere Ausprägungsformen sind bspw. Anteilsrelationen einer Dimension Geschäftsfeld, Plan-Ist-Abweichungen für eine Dimension Version, oder Kennzahlvergleiche in einer Dimension Branche. Diese Beispiele gehören zwar primär in den Bereich der Modellanalyse, weisen jedoch strukturell die gleichen Merkmale wie Konsolidierung auf. Es bietet sich daher an, sie durch ein einheitliches Konstrukt im DSS-Werkzeug zu unterstützen. Dieses Konstrukt muß prinzipiell zwei Teilschritte leisten:

a) Die Definition der hierarchischen Struktur im Sinne der Zusammengehörigkeit von Dimensionsausprägungen; hierdurch werden praktisch die jeweiligen Gültigkeitsbereiche für rechnerische Beziehungen der jeweiligen Dimension festgelegt. Deren Vorhandensein ist allerdings nicht zwingend; so kann bspw. eine Dimension Berichtstyp oder Berichtszeitpunkt auch als Basis für ein Drill-down heterogener Informationsobjekte dienen.

b) Die eigentliche Definition der (rechnerischen) Beziehungen zwischen den Elementen der Hierarchieebenen.

Ähnlich wie bereits bei der Dimensionsspezifikation, wo Merkmalsdefinition und Merkmalszuordnung in vielen Werkzeugen fest gekoppelt waren (s. vorheriges Kapitel), weisen auch die angebotenen Konstrukte zur Konsolidierung wieder zumeist eine nicht auftrennbare Integration dieser Teilschritte auf. Der Berechnungsoperator der Beziehungsdefinition legt zugleich die "Hierarchie"-Beziehung fest. Bei DSS-Generatoren wie OneUp oder FCS-Multi ist der Berechnungsoperator auf die Addition beschränkt. Dies erweist sich bereits als restriktiver Faktor bei üblichen Konsolidierungsanwendungen, z.B. einer Holding, wenn Unternehmensanteile berücksichtigt werden müssen, Unternehmensergebnisse also nur entsprechend der prozentualen Beteiligung in die Addition eingehen dürfen. Generellere Ansätze, die nicht primär auf Konsolidierung ausgerichtet sind, wie z.B. CA-Compete, erlauben das gesamte Spektrum der Beziehungsspezifikation incl. des Einsatzes von Funktionen.

Dabei gelten die Beziehungen entweder aufgrund des immanenten vektoriellen Prinzips (SystemW/OneUp, FCS-Multi) von vorneherein für alle Ausprägungen aller anderen Modelldimensionen, insbesondere also für alle Variablen, Zeitpunkte usw., oder sie können entsprechend globalisiert werden (CA-Compete).

In der Praxis ist dies jedoch nur in Ausnahmefällen angebracht. Selbst reine Konsolidierungsmodelle, z.B. eine Konzernbilanz, weisen Variablen mit Kennzahlencharakter auf, die nicht einfach addiert werden können, z.B. Eigenkapitalrendite. Zur Lösung dieses Problems werden zwei Wege verfolgt. Zum einen Konstrukte zur Einschränkung des Gültigkeitsbereichs der Hierarchiebeziehung im allgemeinen bzw. Konsolidierungsbeziehung im speziellen, zum anderen die Redefinition von "fälschlicherweise" addierten Modellwerten. Das zweite Konzept wird in FCS-Multi verfolgt, indem die Berechnungslogik einen eigenen Anweisungsblock erhält, der nach der (obligatorischen) Addition ausgeführt wird.

Produkte wie CA-Compete vermeiden diese partielle Doppelberechnung durch explizite Spezifikation des Gültigkeitsbereichs. Dies kann individuell für alle anderen Dimensionen geschehen, innerhalb jeder Dimension aber nur entweder für eine oder alle Ausprägungen. Fallunterscheidungen für Gruppen von Ausprägungen müssen daher aufwendig durch Einzelspezifikationen jedes Gruppenelements realisiert werden. Ohne Ergänzung der Globalisierungsfunktion um Ausprägungsgruppen erweist sich dieses Konzept nur für einfache, homogene Problemstrukturen brauchbar; in allen anderen Fällen produziert es Intransparenz und birgt erhebliche Gefahren für die Modellvalidität, da die einer Modellzelle zugeordnete Berechnungsbeziehung von der Reihenfolge der Globalisierung abhängt und bestehende, evtl. gewollte Globalisierungen einer Richtung durch nachträgliche Globalisierungen bezüglich anderer Dimensionen überschrieben werden.

Ein modifiziertes Einschränkungskonzept in Form einer Prioritätenregelung wird bei SystemW/OneUp verfolgt, allerdings nicht von Seiten der jeweiligen Hierarchiedimension aus, sondern umgekehrt durch entsprechende Kennzeichnung der Variablen. Konsolidierung stellt dabei eine Ausprägung des für DSS-Generatoren einzigartigen Konstrukts der Variablentypen dar, mit dem auch weitere Spezialfälle der Modellberechnung effizient spezifiziert werden können und das deshalb im nächsten Kapitel gesondert behandelt wird. Die Beschränkung der Typisierung von Dimensionsausprägungen auf die Variablendimension impliziert allerdings, daß Ausschlüsse der Konsolidierung bezüglich anderer Dimensionen explizit durch Fallunterscheidungen in deren Berechnungslogik spezifiziert werden müssen. Beispiele hierfür sind etwa Anteilsausprägungen wie Quartal in Prozent zum Jahr bei der Zeitdimension oder Produkt in Prozent zur Produktgruppe bei der Produktdimension usw..

Weist die Einführung von Variablentypen bei SystemW/OneUp bereits in die Richtung getrennter Konstrukte für die Spezifikation von hierarchischer Struktur und Berechnungsbe-

ziehung, so ist diese Trennung in macControl konsequent vollzogen. Hierarchie-Strukturen werden durch das Konstrukt der Zuordnungen gebildet, wobei jede neue Hierarchieebene als eigene Merkmalsgruppe definiert werden kann, wenn sich ihre Berechnungsvorschrift von der anderer Ebenen wesentlich unterscheidet. Damit sind Basiselemente über den jeweiligen Oberbegriff lediglich selektierbar, über ihre rechnerische Verknüpfung ist noch nichts ausgesagt. Diese wird erst in einem zweiten Schritt als explizite Summenformel festgelegt, wobei sich die Summierung über die Zuordnung erstreckt und über deren Merkmalsgruppe verallgemeinert wird. Nach dem gleichen Prinzip kann für jede Modelldimension verfahren werden, z.B. können so alle "addierfähigen" betriebswirtschaftlichen Größen in einer Zuordnung zusammengefaßt und diese für die Verallgemeinerung verwendet werden. Allgemein ist für jede identisch berechnete Teilmenge einer Dimension je eine Zuordnung zu bilden, und für jede Kombination dieser Zuordnungen die entsprechende Verknüpfungsformel. Der Vorteil dieses Konzepts ist in seiner maximalen Flexibilität zu sehen, der Nachteil liegt darin, daß keinerlei automatische Standards für einfache "Normalfälle" existieren. Konsolidierung muß also immer mindestens mit einer Rechenformel explizit spezifiziert werden.

Welches Konzept bzw. welche Konstruktkombination im Einzelfall effizienter für die Modellbildung und damit auch -wartung ist, hängt damit entscheidend von der abzubildenden Problemstruktur ab. Als Bewertungskriterium sollten Redundanzfreiheit und Transparenz gewählt werden, d.h. gleich berechnete Modellwerte sollten nur einmal spezifiziert sein, deren Gültigkeitsspektrum muß leicht erkenn- und änderbar sein, umgekehrt müssen aber auch von jedem Modellwert ausgehend Fallunterscheidungen bezüglich zugehöriger Dimensionen erkennbar sein. Konstrukte des DSS-Generators können dies nur fördern, sie müssen durch entsprechende Schulung und Disziplin des Modellentwicklers auch zum Einsatz gebracht werden.

6.2.1.3. Variablentypen

In den meisten DSS-Generatoren bzw. Planungssprachen besitzen die Dimensionsobjekte (Ausprägungen) keine weiteren Attribute außer dem Namen. Somit können auch keine strukturellen oder funktionalen Spezialisierungen realisiert werden. Eine Ausnahme bilden die Produkte SystemW/OneUp. Sie bieten das Konstrukt der Variablentypen an, mit dem zwei Attribute gesetzt werden können, die von der Modellberechnungskomponente ausgewertet werden:

a) CURRENCY/NON CURRENCY: Dieses Attribut stellt einen logischen Schalter dar, mit dem eine Variable als Währungsgröße deklariert werden kann. Sofern in einem gesonderten Spezifikationsteil Währungsparitäten definiert wurden, werden alle Währungsgrößen automatisch entsprechend der gewählten Währung konvertiert. Dies gilt sowohl für die Ausgabe von Modellergebnissen als auch für die Umrechnung im Rahmen der Konsolidierung, falls Dimensionsausprägungen unterschiedlicher Währungen zusammengefaßt werden sollen.

b) CONSOLIDATE/RECALCULATE/OVERRIDE/RATIO/COMMON: Während die Währungsparitäten der übrigen Modellogik nachgelagerte Berechnungen darstellen, nimmt diese Attributgruppe unmittelbaren Einfluß auf die originäre Modellberechnung hierarchisierter Modelle. Der (Standard-)Wert CONSOLIDATE bewirkt automatisch auf allen aggregierenden Modellebenen die Addition der entsprechenden Werte untergeordneter Dimensionsausprägungen anstelle der Anwendung einer evtl. vorhandenen Rechenregel. RECALCULATE/OVERRIDE dagegen bewirken, daß die für die Zelle definierte Rechenregel bzw. der Eingabewert verwendet werden. RATIO stellt einen Unterfall von RECALCULATE dar, der im Zusammenhang mit dem Konstrukt der Regeltypen (s. Kapitel 6.2.1.5) steht: es sorgt dafür, daß die Variablenregel nicht nur im Falle der Konsolidierung sondern auch in Bezug auf Spaltenverknüpfungen, z.B. Summen über die Zeit, Vorrang gewinnt; RATIO erspart dem Modellentwickler damit explizite Fallunterscheidungen bei der Spaltenregel, wie sie typischerweise für als Quotienten gebildete Kennzahlen notwendig wären. COMMON schließlich legt die Variable als Vektor über die Zeit an. Dies stellt die einzige Durchbrechung des Prinzips der vollen Mehrdimensionalität dar, kann aber nur für zeitvariante Eingabewerte genutzt werden, die in allen übrigen Dimensionen identisch sind.

6.2.1.4. Periodentypen

Betriebswirtschaftliche Modelle enthalten bezüglich der Zeitdimension häufig Vergangenheitswerte, für die keine Berechnungen benötigt werden. Beispiele sind Zeitreihen als Basis für gleitende Durchschnitte und Prognosen oder Werte von Vorperioden im Sinne einer rollierenden Planung. Um im Zusammenhang mit der vektoriellen Beziehungsspezifikation den Entwickler von Einschränkungen des Geltungsbereichs von Regeln zu entlasten, wurde in den Planungssprachen SystemW/OneUp das Konstrukt des Periodentyps eingeführt. Dabei wird unterschieden in FORECAST-Perioden, auf die die Rechenregeln angewandt werden, und HISTORY-Perioden, die rechen-neutral sind und vom Zeitpunkt Null an rückwärts gezählt werden. Durch ein Kommando ROLL-DATA wird das für die rollierende Planung typische Verschieben der gesamten Datenmatrix um n Perioden komfortabel und ohne weiteren Spezifikationsaufwand unterstützt.

6.2.1.5. Regeltypen

Die meisten DSS-Generatoren arbeiten mit nur einer Ebene für die Modellbeziehungen (Regeln), d.h. alle rechnerischen Zusammenhänge sind gleichwertig, egal auf welche Dimensionen sie sich beziehen. Das hat zur Folge, daß je Modellwert nur eine Regel definiert werden kann und darf, und bedeutet weiterhin, daß bei über Dimensionsbereiche verallgemeinerten Regeln zahlreiche Unterfälle explizit definiert werden müssen. Ein Teil dieser Fallunterscheidungen läßt sich durch das Konstrukt der Regeltypen in SystemW/OneUp vermeiden. Prinzipiell existieren drei Typen von Regeln: individuell spezifizierbare für die Basisdimensionen Variable (Zeilen) und Zeit (Spalten) sowie fix additive für alle übrigen (Konsolidierungs-)Dimensionen. Die Eindeutigkeit je Modellzelle wird durch eine fest installierte Prioritätenregel realisiert, die Konsolidierung vor Spalte vor Zeile vor Eingabewert lautet. Konsolidierung wird über Typ und Dimensionshierarchie geschaltet, Zeile und Spalte über deren Existenz bzw. den Variablentyp RATIO (s. Kapitel 6.2.1.3).

Nachteile dieses Konzepts ergeben sich vor allem aus der inkonsequenten, nur partiellen Umsetzung, d.h. der Beschränkung auf additive Verknüpfung ab der dritten Dimension, sowie die Tatsache, daß die Prioritätenreihenfolge nicht verändert werden kann. Das bedeutet, daß von der Konsolidierung abweichende Berechnungen mittels aufwendiger, redundanter Fallunterscheidungen in die Ebene der Variablen- oder Spalten-Regeln "projiziert" werden müssen.

6.2.1.6. Einheiten

Modellwerte müssen zu Analysezwecken häufig zu einer Bezugsgröße in Beziehung gesetzt werden, z.B. Differenz bzw. prozentuale Änderungsrate zum Vorjahr, Modellwerte pro Stück oder Produkt- bzw. Regions-Anteile am Gesamtunternehmen usw.. Dies kann theoretisch durch Einrichtung einer zusätzlichen Dimensionsausprägung, evtl. in einer weiteren Dimension mit entsprechender Regeldefinition geschehen. Da es sich aber lediglich um eine temporäre Sicht zu Analyse- oder Darstellungszwecken handelt, ist die resultierende Modellaufblähung überflüssig. Zur Vermeidung dieses Nachteils verfügt das Produkt CA-Compete über ein eigenes Konstrukt *Einheiten*. Mit ihm können quasi berechnete Modellsichten generiert werden. Die Berechnungen können bezüglich aller Modelldimensionen individuell beschränkt werden, im Gegensatz zur Lösung mit zusätzlichen Dimensionen aber nicht dimensionsmäßig differenziert werden, was für diese spezielle Anwendung auch nicht notwendig ist. Dieses Konstrukt leitet als berechnete Modellsicht über zu Konstrukten der Modellanwendung, auf die im folgenden eingegangen wird.

6.2.2. Anwendungs-Spezifikation

6.2.2.1. Modellsichten (Views)

Anwendungen von DSS-Modellen bestehen in ihrer einfachsten Form aus der Selektion und Aufbereitung von Modellteilen zu Zwecken der Werteingabe bzw. -änderung oder (präsentationsreifen) Ausgabe. Die Selektion von Modellteilen kann sich an funktionalen Erfordernissen orientieren, z.B. als Eingabemaske für aktuelle Ist- oder Planwerte, oder organisatorisch determiniert werden, z.B. als Ausgabe Produktbezogener Informationen für den jeweiligen Produktmanager, oder eine Kombination beider Kriterien darstellen, z.B. als Abweichungsanalysebericht je Region für den jeweiligen regionalen Verkaufsleiter. Modellsichten (Views) stellen ein mächtiges Konstrukt in DSS-Generatoren dar, diese alternativen Zusammenstellungen von Modellwerten redundanzfrei, d.h. ohne Mehrfachspeicherung von Werten und Beziehungen, und effizient zu realisieren. Die Leistungsfähigkeit des View-Konstrukts kann nach folgenden Kriterien klassifiziert werden:

a) View-Bibliothek

Das View-Konstrukt impliziert die Möglichkeit, alternative Sichten erzeugen, benennen und als Auswahlliste bereitstellen zu können, wie dies in Abb. 6-3 am Beispiel von CA-Compete dargestellt ist. Produkte wie SystemW/OneUp erlauben dem Anwender zwar auch das beliebige Vertauschen von Modelldimensionen bzw. Sortieren und Ausblenden von Ausprägungen, resultierende Sichten lassen sich aber nur in Verbindung mit dem Reportgenerator oder der ExecuView-Komponente des EIS-Generators CommanderEIS speichern (vgl. Kapitel 5.3.2). In Spreadsheet-Paketen wie MS-Excel werden Views als Verknüpfung von Zellbereichen in eigenständigen Fenstern realisiert, die allerdings nur im Rahmen der MS-DOS-Dateinamenskonventionen (8 Stellen) benannt werden können.

b) View-Funktionalität

Dieses Kriterium beschreibt die Möglichkeiten, Views strukturell und inhaltlich manipulieren zu können, sowie die Abhängigkeiten zwischen Views. Jede nicht nur lesende Funktionalität setzt eine Trennung zwischen Modellspezifikation einerseits und Modellpräsentation andererseits voraus. Während in klassischen Spreadsheetpaketen wie MS-Excel die Modellspezifikation mit zumindest einem "View" identisch ist, generieren mehrdimensionale Planungssprachen wie SystemW/OneUp, CA-Compete oder macControl ein eigenständiges Hintergrundmodell, das strukturell und inhaltlich von beliebig vielen, weiteren Views modifiziert werden kann. D.h., sowohl Modellwerte als auch -beziehungen oder gar Dimensionsausprägungen bzw. ganze Dimensionen können mit globaler Wirkung für alle anderen Views von jedem View aus geändert, ergänzt

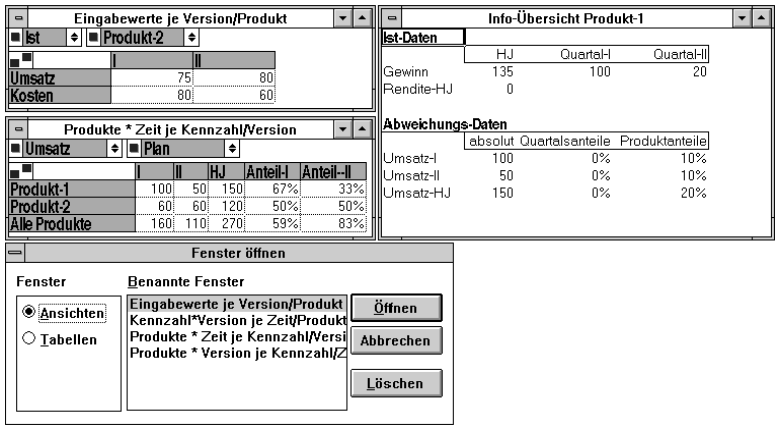


Abb. 6-3:	Beispiele des View-Konstrukts in CA-Compete

oder gelöscht werden. Im Falle von MS-Excel müssen derartige Modifikationen aufgrund der rein lesenden Verknüpfung zu anderen "Views" grundsätzlich im Ursprungs-"View" erfolgen. Sie haben daher reinen Reportcharakter und sind den Reportgenerator-Komponenten von Planungssprachen gleichzusetzen.

c) Dimensionale Flexibilität

Darunter soll die Möglichkeit verstanden werden, in einer Sicht Modelldimensionen zu mischen, d.h. in einer View-Spalte oder -Zeile Ausprägungen verschiedenster Modelldimensionen anordnen zu können. Ein Beispiel hierfür ist etwa die Kombination von Plan-, Ist- und Abweichungswerten mit ausgewählten betriebswirtschaftlichen Kennzahlen je Quartalswerten mit deren Jahresanteilen. Bei voll mehrdimensionalen DSS-Generatoren bzw. Planungssprachen, die das Prinzip des kartesischen Dimensionsprodukts verfolgen, ist hierzu auf Masken- oder Report-Generatoren zurückzugreifen: CA-Compete bietet hierfür den Fenstertyp Tabelle (vgl. Abb. 6-3), SystemW einen Maskengenerator, OneUp die (eigenständige) Systemkomponente OneUP-Builder. In macControl sind Modell- und Anwendungsspezifikation im Konstrukt der *Arbeitsblätter* integriert, so daß es unmittelbar möglich ist, Views beliebiger Dimensionskombinationen während der Modellspezifikation herzustellen, die somit über volle, globale Änderungsfunktionalität verfügen.

6.2.2.2. Ursachenanalyse

Ein wesentlicher Beweggrund für die Erstellung von DSS liegt in der modellbasierten Untersuchung von Art und Ausmaß der Zusammenhänge realer Größen. Nach dem verfolgten Hauptzweck werden Modelle üblicherweise in Beschreibungs-, Erklärungs- und Entscheidungsmodelle unterschieden (vgl. [Stachowiak 73, Modelltheorie]). Im Gegensatz zu reinen Beschreibungsmodellen müssen Erklärungsmodelle als Basis für eine aussagekräftige Ursachenanalyse vermehrt kausale Ursache-Wirkungs-Beziehungen in Form empirisch fundierter oder hypothetisch formulierter Relationen abbilden. Darauf aufbauend kommen spezielle DSS-Funktionen (Konstrukte) zur Ursachenanalyse zur Anwendung, deren Funktionalität nach dem Kriterium der Aussagekraft wiefolgt klassifiziert werden kann:

a) Identifikation statischer unmittelbarer Einflußgrößen

b) Bewertung und Rangordnung von Einflußgrößen in Bezug auf ihren Beitrag zur Erklärung der Dynamik zu erklärender Größen, d.h. den Grad der Abhängigkeit der Wertänderung abhängiger Größen von Wertänderungen ihrer Einflußgrößen.

c) Identifikation statischer oder dynamischer Wirkungsketten bis hinab zu unabhängigen Modellgrößen. Dabei kann es sich bspw. entweder um durch das Unternehmen mittelbar bzw. unmittelbar beeinflußbare Entscheidungsvariablen, z.B. Lohnkosten bzw. Produktionsmengen oder um als Randbedingung restriktiv zu beachtende, exogene Umweltfaktoren, z.B. Steuersätze oder Altersstruktur potentieller Konsumenten, handeln.

Die einfachsten Konstrukte von DSS-Generatoren zur Unterstützung einer Ursachenanalyse beschränken sich auf die Anzeige der unmittelbaren Einflußgrößen, abgeleitet aus den Modellgleichungen. Ihre Aussagekraft nimmt mit steigender Komplexität der eingehenden Modellbeziehungen stark ab, z.B. bei Fallunterscheidungen in Form von Preis-Absatz-Funktionen oder Aggregationsbeziehungen in mehrdimensionalen Modellen. Ein Beispiel hierfür stellt die WHAT-AFFECTS-Funktion von OneUp dar, die sich ausschließlich auf prinzipielle Abhängigkeiten der Variablendimension beschränkt. Weitergehende Ansätze wie z.B. die Ableitungs-Funktion in CA-Compete orientieren sich an einzelnen Modellzellen und detaillieren nach deren Dimensionsausprägungen, gehen aber über eine reine statische Anzeige der Modellbeziehungen und daraus resultierender aktueller Werte auch nicht hinaus (vgl. Abb. 6-4).

Das bislang weitestgehende Konstrukt zur Ursachenanalyse in DSS ist durch die EXPLAIN-Funktion in der Planungssprache IFPS/Plus gegeben (vgl. [Rieger 90, Wissensbasierte Planungssprachen], S. 259 ff.). Das Spektrum unterstützter Fragestellungen reicht von einfachen Modellzusammenhängen, z.B. *"Wie ergibt sich RENDITE?"*, über Periodenvergleiche,

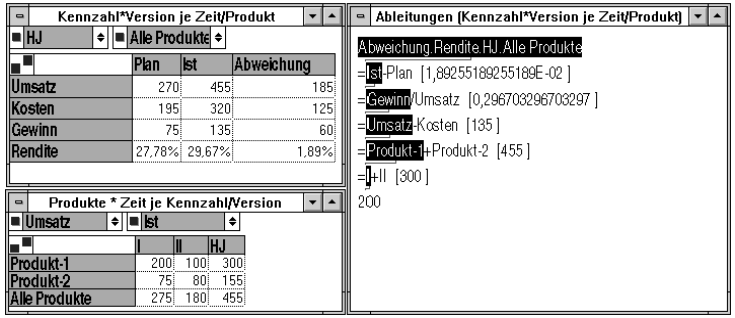


Abb. 6-4:	Beispiel-Konstrukt zur Ursachenanalyse in der Planungssprache CA-Compete

z.B. *"Warum sinkt Gewinn in 1992?"* bis zu Szenarienvergleichen (vgl. Kapitel 6.2.2.5), z.B. *"Warum sind die Kosten in Fall B größer als in Fall A?"*. Die generierten Antworten basieren auf einer systematischen Variation aller Einflußgrößen einer Modellzelle mit entsprechender Durchrechnung des Modells. Sie werden dann bezüglich ihres Signifikanzgrades für die Fragestellung bewertet und in eine Rangordnung gebracht, die Bestandteil und Basis einer aus Textkonserven zusammengestellten, an die natürliche Sprache angelehnten Antwort (Erklärung) ist (vgl. Abb. 6-5). Allerdings sind komplexe Modellbeziehungen, z.B. Fallunterscheidungen oder Simultangleichungen derzeit nicht auflösbar. Ferner liegen die entscheidenden Parameter und "Regeln" des induzierten Sensitivitätsanalyse- und Bewertungsprozesses für Modellentwickler und -Anwender unbeeinflußbar im Dunkeln.

6.2.2.3. What-if-Analyse

Die im vorangegangenen Kapitel beschriebene EXPLAIN-Funktion impliziert ein Konstrukt, das als "What-if-" bzw. Sensitivitätsanalyse zu den elementaren Bestandteilen von DSS-Generatoren zu zählen ist (vgl. [Reimann 85, DSS-Software]). Als Klassifizierungskriterien für die Leistungsfähigkeit von Konstrukten zur Sensitivitätsanalyse kommen in Betracht:

Anfrage-Spezifikation

```
MODEL: TEST
  COMMANDS          VARIABLES           DID COMPARISONS      COLUMNS
  WHY DID           MARKET GROWTH IN    GO UP                1987
  WHY IS            COST OF GOODS IN    GO DOWN              1988
  DEFINITION FOR    R AND D INDEX       CHANGE               1989
  VALUE OF          COST OF SALES IN    STAY THE SAME        1990
  DIAGRAM           INTEREST INDEX      GO UP SO MUCH        1991
  OUTLINE           TAX INDEX           GO UP SO LITTLE      1992
  AGAIN             SALES               GO DOWN SO MUCH      1993
  CONTINUE          COST OF GOODS SO    GO DOWN SO LITTLE    1994
  CONTINUE DOWN     GROSS PROFIT        PEAK                 1995
  MSGWAIT           RESEARCH AND DEV    DIP                  1996
  QUIT              SALES EXPENSE                            1997
                    INTEREST EXPENSE    IS COMPARISONS
                    DEBT                HIGHER
                    TOTAL EXPENSES      LOWER
                    PROFIT BEFORE TA    THE SAME AS
                    TAXES

===> WHY DID SALES EXPENSE GO UP IN 1990

PF: 1 -HELP     2 -CMD      3 -EXIT     4 -DEL      5 -VALUE     6 -REDO
    7 -PG UP    8 -PG DN    9 -TOP      10-BOT      11-FIND      12-SORT
```

Antwort des EXPLAIN-Konstrukts

```
MODEL: TEST
*************************************************************************
SALES EXPENSE WENT UP IN 1990 BECAUSE SALES INCREASED. THIS RESULT WAS
PARTIALLY OFFSET BECAUSE COST OF SALES INDEX DECREASED.

SALES EXPENSE            = SALES * COST OF SALES INDEX / 100

                                        AMOUNT
                                1989    CHANGE      1990
                              --------  --------  --------
SALES EXPENSE                   14.30       .10     14.40
SALES                           55.00      5.00     60.00
COST OF SALES INDEX             26.00     -2.00     24.00









********************** <ENTER> - RETURN TO MENU ********************
WHY DID SALES EXPENSE GO UP IN 1990
```

Abb. 6-5:	Wissensbasierte Ursachenanalyse mit dem EXPLAIN-Konstrukt in IFPS/Plus

a) Die Zahl gleichzeitig beobachtbarer Modellgrößen
b) Die Zahl gleichzeitig veränderbarer Modellgrößen
c) Die Zahl gleichzeitig beobachtbarer Änderungen

In der einfachsten Form ist es lediglich möglich, die Auswirkung der Änderungen einzelner oder mehrerer, in der Regel unabhängiger Modellgrößen auf eine oder mehrere, ausgewählte abhängige Modellgrößen temporär ermitteln zu lassen. Als Hilfsmittel zur Entscheidungsvorbereitung sind jedoch leistungsstärkere, aussagekräftigere Konstrukte notwendig, die insbesondere einen Alternativenvergleich ermöglichen, d.h. die Auswirkungen verschiedener Änderungen einer oder mehrerer Modellgrößen gestatten.

Abb. 6-6 zeigt eine derartige Anwendung am Beispiel der Datentabelle-Funktion von CA-Compete, die eine Variation von maximal zwei beliebigen Modellzellen in individuellen Intervallen und die Anzeige maximal einer, ebenfalls beliebigen Modellzelle erlaubt. Im Beispiel werden die Auswirkungen einer Variation der Ist-Kosten des Produkts-2 im Quartal-II von 100 bis 200 in Schritten zu 20 in Kombination mit der Variation des Ist-Umsatzes von Produkt-1 im Quartal-I auf die Plan-Abweichung der Rendite des 1.Halbjahres für alle Produkte dargestellt. Die Ergebnisse werden automatisch in einer eigenen, dynamischen Modellsicht zusammenfassend als Tabelle dargestellt.

Sensitivität							
	I.Kosten.Ist.Produkt-1	100	120	140	160	180	200
II.Umsatz.Ist.Produkt-2	HJ.Rendite.Abweichung.Alle Produkte						
100		4,85%	0,64%	-3,57%	-7,78%	-11,99%	-16,20%
120		7,58%	3,54%	-0,51%	-4,55%	-8,59%	-12,63%
140		10,09%	6,20%	2,32%	-1,56%	-5,45%	-9,33%
160		12,41%	8,67%	4,93%	1,19%	-2,54%	-6,28%
180		14,56%	10,96%	7,36%	3,75%	0,15%	-3,45%
200		16,57%	13,09%	9,61%	6,14%	2,66%	-0,82%

Abb. 6-6: Beispiel eines What-If-Konstrukts in CA-Compete

Noch leistungsfähigere, Großrechnerbasierte Planungssprachen wie SystemW, FCS oder IFPS/Plus gestatten insbesondere die Analyse von mehr als einer beobachteten Modellgröße, z.B. den für die Entscheidungsvorbereitung besonders wichtigen Aspekt einer Größe über die Zeit. Ihre Anwendung wird jedoch zumeist dadurch erschwert, daß das Konstrukt einerseits nur als parametrisiertes Kommando zur Verfügung steht bzw. die Ergebnisausgabe rein tabellarisch auf den Bildschirm oder in eine Datei erfolgen kann. Für eine effektive Nutzung sind vor- bzw. nachgelagerte Makros resp. Prozeduren zu erstellen, die eine komfortable Parametereingabe und leicht interpretierbare, in der Regel graphische Ausgabe ermöglichen.

6.2.2.4. Goalseeking

Die Konstrukte der Sensitivitätsanalyse können in zweierlei Hinsicht als Unterstützung der Problemlösungsphasen der Alternativengenerierung und -bewertung interpretiert werden:

a) Zum einen stellen sie mit der (systematischen, schrittweisen) Variationsmöglichkeit unabhängiger Modellgrößen die Aktionsvariablen und -intervalle einer Alternativengenerierung zur Verfügung.

b) Zum anderen verfügen sie mit der Messung der Auswirkungen jeder (neuen) Aktionsvariablen-Konstellation über die Bewertungsgrundlage bezüglich deren Zielbeitrag.

Die Unterstützungsleistung ist jedoch nicht zielgerichtet. Sowohl die Alternativensuche als auch die Bewertung der Alternativen untereinander obliegen der Steuerung durch den Anwender. Konstrukte, die auch diese Funktionen übernehmen, sind unter den Begriffen Optimierung und Zielsuche (Goalseeking) bzw. What-to-do-to-achieve bekannt. Die Leistungsfähigkeit dieser Konstrukte hängt von den Ausgestaltungsformen folgender Grundelemente jeder Zielrechnung im weiteren Sinne ab:

- den *Variablen*, d.h. insbesondere wieviele und welcher Typ von Aktionsvariablen berücksichtigt werden können,
- den *Restriktionen*, d.h. insbesondere, ob Wertintervalle für die Aktionsvariablen selbst vorgegeben und ob bzw. welche Art von Bedingungen zwischen den Aktionsvariablen definiert werden können, sowie
- der *Zielfunktion*, d.h. wie die Zielgröße definiert und welches Zielkriterium für den Suchprozeß vorgegeben werden kann.

Nachfolgend werden die verschiedenen Ausprägungen dieser Elemente exemplarisch anhand der Zielsuche-Konstrukte repräsentativer DSS-Generatoren dargestellt.

- *CA-Compete*

  Das Konstrukt Zielsuche in der Planungssprache CA-Compete gestattet in der Version 4.2 nur eine Variable vom Typ Eingabe, d.h. ohne Berechnungsformel. Abb. 6-7 zeigt eine Zielsuche-Spezifikation anhand des bereits vorher verwendeten Beispielmodells: Dabei soll der Ist-Umsatz von Produkt-I im 1.Quartal so eingestellt werden, daß die Rendite-Abweichung gesamt zum Plan für das 1. Halbjahr gleich Null wird.

  Für die (eine) Aktionsvariable kann dabei ein Wertebereich per Ober- und Untergrenze festgelegt werden. Als "Zielfunktion" kann ebenfalls nur ein Modellwert angegeben werden, der sich direkt oder indirekt aus der Variablen berechnet. Dies stellt jedoch keine gravierende Einschränkung dar, da jeder berechnete Modellwert verwendet wer-

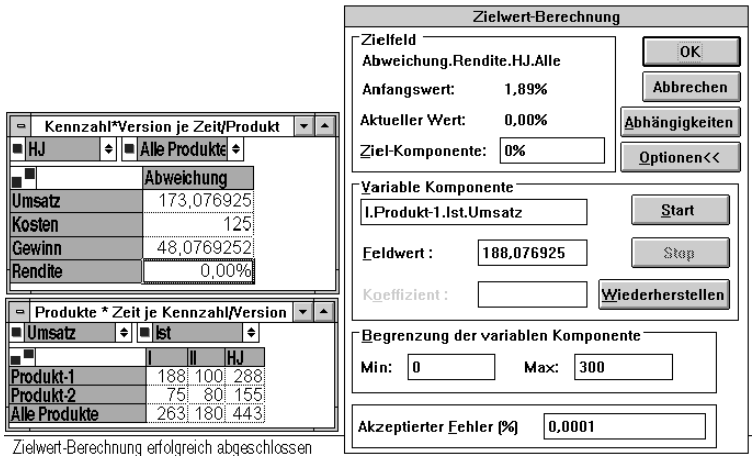


Abb. 6-7:	Das Zielsuche-Konstrukt in CA-Compete

den kann. Die eigentliche Zielfunktion wird somit implizit als Formel im Modell realisiert. Ihr Typ ist lediglich durch den Verknüpfungs- und Funktionsvorrat zur Formelspezifikation beschränkt, kann also auch nicht-linear sein. Als Zielkriterium kann jedoch nur das Erreichen eines festen Vorgabewertes der Zielfunktion, nicht etwa deren Maximierung oder Minimierung definiert werden. Insofern reduziert sich die "Zielfunktion" implizit auf eine "Gleichheits-Restriktion" für die Variable, wobei der Restriktionswert für die Lösbarkeit innerhalb des Variablenintervalls liegen muß.

In dieser typischen Ausprägung des Zielsuche-Konstrukts von Planungssprachen findet also keine Optimierung, sondern lediglich die Überprüfung der Lösbarkeit einer y=f(x)-Gleichung statt. Um dies für beliebig komplexe Gleichungstypen zu ermöglichen, wird jeweils ein iterativer Suchalgorithmus eingesetzt, der in der Regel eine Konvergenz, d.h. Stetigkeit der Funktion voraussetzt.

❒ *Großrechner-Planungssprachen*

Die leistungsfähigsten Zielsuche-Konstrukte von Planungssprachen sind noch in Großrechner-Versionen zu finden. Das ADJUST-Kommando von SystemW gestattet sowohl mehr als eine Aktionsvariable als auch Extremwertsuche als Zielkriterium (MAX, MIN). Allerdings können keine Intervalle für die Variablen festgelegt werden. Das umfangreichste Konstrukt ist in IFPS/Plus mit der Zusatzkomponente IFPS/OPTIMUM

realisiert. Dabei können außer für die Variablen selbst auch Restriktionen für weitere beliebige, berechnete Modellzellen spezifiziert werden. Diese Parameter für die Optimierung müssen ebenso wie die Spezifikation der Zielfunktion nicht implizit in das eigentliche Modell integriert werden, sondern werden außerhalb in einem eigenen Optimierungsmodell-Bereich vorgenommen.

6.2.2.5. Cases

In allen Fällen der Sensitivitätsanalyse oder Zielsuche werden die Berechnungen auf einer temporären Kopie des Modells ausgeführt. Sie können danach wahlweise ins Modell übernommen oder verworfen werden. Für eine Unterstützung der Problemlösungsphasen Alternativen-Bewertung und -Auswahl ist es jedoch notwendig, diese temporären Ergebnisse "aufbewahren" und untereinander vergleichen zu können. Dies kann grundsätzlich entweder selektiv oder in Form ganzer Alternativ-Modelle geschehen.

Der erste Fall (selektiv) kann als sammelnde Report-Funktion interpretiert werden. Das oben ausgeführte Datentabelle-Konstrukt der Sensitivitätsanalyse-Funktion von CA-Compete stellt ein Beispiel hierfür dar. Es handelt sich dabei jedoch nur um wertmäßige Modell-Alternativen, d.h. die Modellstruktur ist weitestgehend identisch. Alternative Modellstrukturen können höchstens durch Modellvariablen mit Schalter-Charakter implizit in das (eine) Modell integriert werden: so könnten verschiedene Bestellmengen-Variablen jeweils unterschiedlicher Bestellpolitik (-formel) definiert und über multiplikative Binär-Variablen so miteinander verknüpft werden, daß jeweils nur eine Variante zur Wirkung kommt. Die Verwendung dieses Konstrukts im Rahmen der Zielsuche setzt allerdings einen gemischt-ganzzahligen Optimierungs-Algorithmus voraus, der in herkömmlichen Planungssprachen nicht verfügbar ist. In diesem Falle hilft nur eine Kopplung mit spezialisierten DSS-Generatoren zur Optimierung, z.B. MPSX/MIP von IBM weiter.

Der zweite Fall (Alternativ-Modelle) entspricht einem neuen Konstrukt, das als Cases bekannt ist. Hierbei werden auch strukturell veränderbare Kopien des Ursprungsmodells angelegt, d.h. es können sowohl neue Modellvariablen als auch Modellbeziehungen eingeführt bzw. bestehende Formeln strukturell verändert werden. Im Beispiel der Planungssprache IFPS/Plus können sowohl vergleichende Ursachenanalysen zwischen diesen Fällen (Cases) durchgeführt als auch vergleichende Berichte ausgewählter Modellwerte erstellt werden.

6.2.3. Modell-Berechnung

6.2.3.1. Pull- vs. Push-Prinzip

Die implementierten Konstrukte der Modell-Berechnungs-Komponente von DSS-Generatoren haben unmittelbaren Einfluß auf die Modell- und Anwendungs-Spezifikation und determinieren damit den Entwicklungsaufwand bzw. die Qualität des Entwicklungsergebnisses maßgebend mit. Das einfachste Konstrukt bezieht sich auf das verwendete Prinzip, nach dem Modellwerte berechnet werden. Im Falle des Push-Prinzips, das in heutigen DSS-Generatoren weitestgehend der Vergangenheit angehört, muß der Anwender die Berechnung von Modellwerten selbst aktiv anstoßen. Im Falle des Pull-Prinzips ermittelt der DSS-Generator für die zur Anzeige vom Anwender ausgewählten Modellwerte selbst, ob eine (Neu-)Berechnung notwendig ist und führt diese automatisch durch.

Um Modell-Konsistenz und -Validität zu wahren, müssen dabei auch alle indirekt betroffenen Modellwerte, die nicht angezeigt werden, berechnet werden. Voraussetzung dafür ist, daß der Rechenalgorithmus zellenweise orientiert ist und sich über das gesamte Modell erstreckt, d.h. insbesondere nicht auf eine Hierarchie von Basismodellsichten beschränkt ist, wie dies bei der Planungssprache FCS mit der "Variablen*Zeit-Spreadsheet"-weisen Vorgehensweise der Fall ist. Ferner muß das aus wissensbasierten Systemen bekannte Prinzip der Rückwärtsverkettung zum Einsatz kommen, vereinfacht jedoch dadurch, daß aufgrund eindeutiger Beziehungsspezifikationen konventioneller DSS-Modelle keine alternativen Berechnungswege berücksichtigt werden müssen. Somit entfällt praktisch die Notwendigkeit einer Konfliktlösungs-Komponente. Sind dennoch - wie im Beispiel von OneUp - mehrere Formeln für eine Modellzelle definiert, z.B. als Überschneidung von Konsolidierungs-, Perioden- und Variablenregel oder Eingabewert, dann wird die Rolle des Konfliktlösers durch eine einfache Prioritätsregel übernommen. Abb. 6-8 zeigt diesen universellen Algorithmus, der insbesondere dazu geeignet ist, alle im Kapitel 6.2.3.2 dargestellten, denkbaren Fälle nicht-prozeduraler Modellspezifikation in mehrdimensionalen DSS-Generatoren auflösen zu können.

6.2.3.2. Nichtprozeduralität

Die Möglichkeit, Modell- und Anwendungsspezifikation nach sach-logisch zusammen gehörenden Einheiten statt nach der rechen-logisch notwendigen Reihenfolge vornehmen zu können, erfordert in der Modell-Berechnungs-Komponente Konstrukte zur Bewältigung von *Nichtprozeduralität*. Die Vorteile nicht-prozeduraler Modell-Spezifikation sind in einer höheren Modelltransparenz und Verständlichkeit für den sich zunehmend aus der Fachabteilung rekrutierenden Entwickler zu sehen.

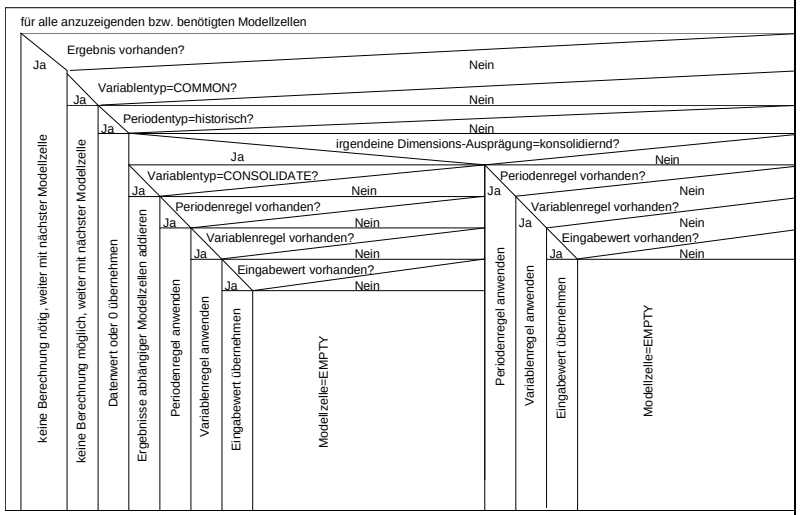


Abb. 6-8: Modellberechnungs-Algorithmus nach dem Pull-Prinzip in SystemW/OneUp

Diesem Fachspezialisten ist danach nicht zuzumuten, sich mit der prozedural richtigen Reihenfolge der Berechnungen, z.B. von Kennzahlen, Verrechnungssätzen usw. herumzuschlagen. Es kann aber nicht übersehen werden, daß es im Zusammenhang mit der Sicherstellung der Modellvalidität häufig nachträglich dennoch notwendig ist, die resultierende prozedurale Reihenfolge der Modellbeziehungen überprüfen zu können. Diese Funktion deckt sich mit der bereits beschriebenen Funktion der Ursachenanalyse (s. Kapitel 6.2.2.2) und kann durch deren Konstrukte, z.B. das dargestellte Ableitungs-Konstrukt adäquat unterstützt werden.

Nichtprozeduralität ist angesichts der Möglichkeiten der Mehrdimensionalität in konventionellen DSS-Generatoren, speziell Planungssprachen, keineswegs trivial. Zum Beispiel müssen zur Umlage von Gemeinkosten auf Kostenstellen zuerst der Gesamtkostenwert der abgebenden Kostenstelle berechnet, die Summe der Verbrauchswerte aller empfangenden Kostenstellen bezüglich der Bezugsgröße ermittelt, danach die individuellen Verbrauchswerte zu dieser Summe in Beziehung gesetzt und mit dem Gesamtkostenwert multipliziert werden. Diese Berechnungskette ist durch Vorwärtsverweise über die Variablen und die Konsolidie-

rungshierarchie, bei mehrperiodischer Betrachtung zusätzlich über die Zeitdimension charakterisiert. Abb. 6-9 zeigt anhand einer Implementierung dieses Beispiels mit der Expertensystem-Shell ESE von IBM die nach dem Pull-Prinzip zu berechnenden Modellzellen sowie deren per Rückwärtsverkettung generierte Berechnungsreihenfolge für die Abfrage von zwei unterschiedlichen Modellzellen, und zwar der Endstellenkosten der Hilfskostenstelle bzw. aller Hauptkostenstellen.

Abfrage: Endstellenkosten Hilfskostenstelle

Ergebniswerte	Hilfskostenstelle	Hauptkostenstelle 1	Hauptkostenstelle 2	Hauptkostenstellen 1+2
Kostenart 1	100	unknown	unknown	unknown
Kostenart 2	50	unknown	unknown	unknown
Primärkosten	150	unknown	unknown	unknown
Sekundärkosten	-150	45	105	150
Endstellenkosten	0	unknown	unknown	unknown
Verbrauchswert	unknown	300	700	1000

Rechenreihenfolge	Hilfskostenstelle	Hauptkostenstelle 1	Hauptkostenstelle 2	Hauptkostenstellen 1+2
Kostenart 1	1	unknown	unknown	unknown
Kostenart 2	2	unknown	unknown	unknown
Primärkosten	3	unknown	unknown	unknown
Sekundärkosten	10	7	8	9
Endstellenkosten	11	unknown	unknown	unknown
Verbrauchswert	unknown	4	5	6

Abfrage: Endstellenkosten Hauptkostenstellen 1+2

Ergebniswerte	Hilfskostenstelle	Hauptkostenstelle 1	Hauptkostenstelle 2	Hauptkostenstellen 1+2
Kostenart 1	100	120	230	unknown
Kostenart 2	50	180	170	unknown
Primärkosten	150	300	400	unknown
Sekundärkosten	unknown	45	105	unknown
Endstellenkosten	unknown	345	505	850
Verbrauchswert	unknown	300	700	1000

Rechenreihenfolge	Hilfskostenstelle	Hauptkostenstelle 1	Hauptkostenstelle 2	Hauptkostenstellen 1+2
Kostenart 1	4	1	12	unknown
Kostenart 2	5	2	13	unknown
Primärkosten	6	3	14	unknown
Sekundärkosten	unknown	10	15	unknown
Endstellenkosten	unknown	11	16	17
Verbrauchswert	unknown	7	8	9

Abb. 6-9: Pull-Prinzip und Nichtprozeduralität am Beispiel der Kostenstellenumlage

Eine Klassifizierung der "Schweregrade" aller möglichen Fälle von Nichtprozeduralität für Planungssprachen, die sich aus deren Mehrdimensionalität ergeben kann, wird in [Rieger 90, Wissensbasierte Planungssprachen], S. 253ff. gegeben.

Das in Abb. 6-10 dargestellte Spektrum läßt sich danach durch folgende drei Kriterien vollständig beschreiben:

❒ *Anzahl betroffener Modelldimensionen*
Dieses Kriterium gibt an, ob sich die "Unordnung" in den Beziehungs-Spezifikationen innerhalb nur einer Modell-Dimension, z.B. der Variablen, abspielt, oder ob andere Modell-Dimensionen, evtl. auch nur partiell, d.h. mit einem Teil ihrer Ausprägungen, betroffen sind. Im ersten Fall genügt ein einfaches Umsortieren der Variablen-Spezifikationen.

❒ *Art der betroffenen Modell-Dimensionen*
Dieses Kriterium differenziert danach, ob nur Basis-Dimensionen, d.h. Variablen und/oder Zeit, einbezogen sind, oder eine Vermischung mit Konsolidierungs-Dimensionen vorliegt, die aus Effizienzgründen gewöhnlich erst in einem nachgelagerten Berechnungsschritt ermittelt werden. Im zweiten Fall muß die Orientierung der Berechnung an Basis-Spreadsheets aufgegeben, zumindest jedoch unterbrochen werden.

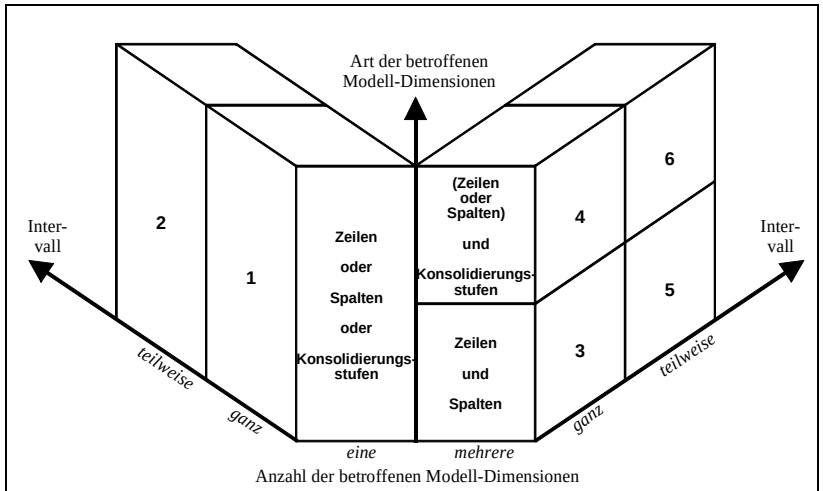


Abb. 6-10: Klassifikation der Nichtprozeduralität in mehrdimensionalen Planungssprachen

❒ *betroffenes Intervall*
Ferner ist eine Differenzierung bezüglich des involvierten "Intervalls" je Dimension zu berücksichtigen, d.h. ob alle Ausprägungen der jeweiligen Dimension oder nur eine Teilmenge davon in gleichem Maße betroffen sind. Bei Teilmengen scheidet ein einfaches Umsortieren der (vektoriellen) Beziehungs-Spezifikationen aus, vielmehr ist zuvor eine Parzellierung des Modellraums vorzunehmen.

Ein Beispiel für den kompliziertesten Fall stellt die bereits mehrfach verwendete durchschnittliche Plan-Abweichung der Rendite für das 1. Halbjahr eines Mehrproduktunternehmens dar: beteiligt sind neben Variablen (Rendite, Umsatz, Kosten) und Zeit noch zwei Konsolidierungs-Dimensionen (Produkt, Version); die Zeitdimension ist dabei nur partiell betroffen (Halbjahr).

Ein über alle Modelldimensionen funktionsfähiger Mechanismus zur automatischen Ermittlung der richtigen Berechnungsreihenfolge - wie ihn der Algorithmus aus Abb. 6-8 nach dem

Pull-Prinzip darstellt - kann den Anwender insofern erheblich entlasten, als dieser nur noch die Logik überschaubarer Teilschritte der Berechnung (nichtprozedural) im Sinne von Kontexten spezifizieren muß.

6.2.3.3. Simultangleichungen

Spezielle betriebswirtschaftliche Zusammenhänge zeichnen sich durch "Ringschlüsse" in den Beziehungs-Spezifikationen aus. Die Fähigkeit der Modell-Berechnungs-Komponente zur Auflösung dieser Simultangleichungen stellt ein weiteres wichtiges Konstrukt in DSS-Generatoren dar. Typisches Beispiel ist das mathematische Verfahren zur innerbetrieblichen Leistungsverrechnung. Zum Einsatz kommen Algorithmen zur Lösung überbestimmter Gleichungssysteme nach Gauß-Seidel oder Aitken. Neben klassischen Spreadsheet-Paketen wie MS-Excel lassen selbst spezialisierte Planungssprachen wie CA-Compete derartige Konstrukte vermissen.

Im übrigen gilt das in Kapitel 6.2.3.2 (Nichtprozeduralität) Gesagte analog, d.h. Simultangleichungen müssen auch über Dimensionsgrenzen hinweg erkannt und aufgelöst werden, wie z.B. in den mehrdimensionalen Planungssprachen SystemW/OneUp.

6.3. Wissensbasierte DSS-Generatoren

6.3.1. Übertragung des Referenzmodells

Für wissensbasierte DSS-Generatoren wurden spezialisierte konzeptionelle Modelle entwickelt, die sich überwiegend an den Architektur-Komponenten und deren Zusammenwirken orientieren, um die Unterschiede zu konventionellen Softwaresystemen herauszustellen. Das bekannteste und allgemein anerkannte ist in Abb. 6-11 dargestellt (vgl. [Puppe 91, Expertensysteme], S. 13). Es besteht aus zwei Hauptkomponenten, die wie folgt zu konventionellen Systemen in Beziehung gesetzt werden können:

❒ *Wissensbasis*

Sie kann nach dem Kriterium der Wissensquelle unterteilt werden in bereichs- und fallbezogenes Wissen von Experten und Anwendern und "berechnetes" Wissen, z.B. Zwischen- und Endergebnisse (vgl. [Puppe 91, Expertensysteme], S. 12). Für eine Analogie zu konventionellen DSS besser geeignet ist das Kriterium des Gebrauchs der Wissensarten. Danach existieren Fakten-, Ableitungs- und Steuerungs- oder Kontrollwissen (vgl. [Puppe 91, Expertensysteme], S. 12). Bezogen auf das Referenzmodell aus Kapitel 6.1 entspricht das Faktenwissen den Modellstrukturen und -daten konventioneller DSS, das

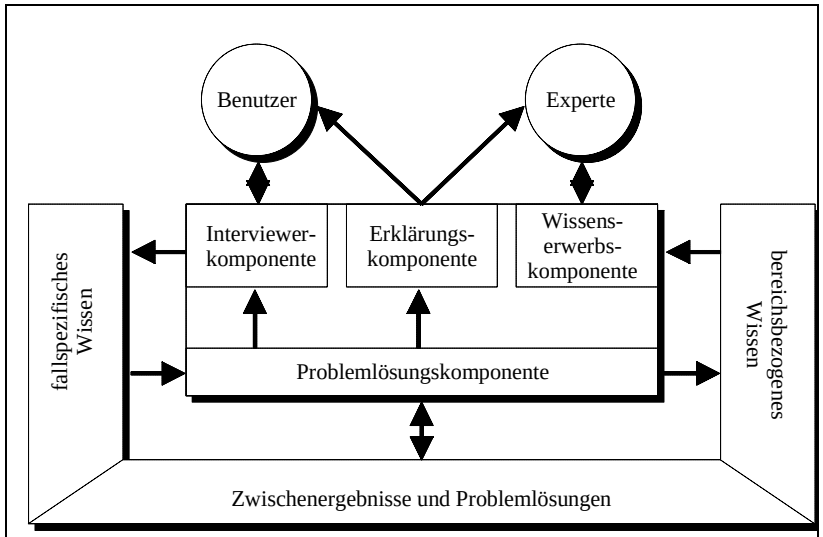


Abb. 6-11:	Architekturmodell von Expertensystemen. (Quelle: [Puppe 91, Expertensysteme], S. 13)

Ableitungswissen den dortigen Modellbeziehungen, z.B. Berechnungsgleichungen. Steuerungs- oder Kontrollwissen dagegen findet sich dort nicht explizit sondern ist fest in die Modellberechnungs-Komponente integriert. Diese Tatsache wird auch häufig als grundlegender Unterschied zwischen konventionellen und wissensbasierten Systemen herangezogen (vgl. [Puppe 91, Expertensysteme], S. 2; [Kurbel 89, Expertensysteme], S. 18). Allenfalls das Konstrukt der Variablentypen (vgl. Kapitel 6.2.1.3) kann als Ansatz eines expliziten Kontrollwissens in konventionellen DSS aufgefaßt werden.

❒ *Steuersystem*

Das Kernstück des Steuersystems wird von der Problemlösungs- oder Inferenzkomponente gebildet. Sie entspricht quasi dem verbleibenden Rest der konventionellen Modell-Berechnungs-Komponente nach Auslagerung von Teilen des Steuerungs- und Kontrollwissen. Infolge dieser Auslagerung weist diese Inferenzkomponente wissensbasierter DSS im Vergleich zur Modell-Berechnungs-Komponente konventioneller DSS ein höheres Maß an Anwendungsunabhängigkeit aus.

Während die Interviewer-Komponente weitestgehend der Dialog-Schnittstelle konventioneller DSS entspricht, hebt sich die Wissensakquisitions-Komponente von ihrem

Äquivalent der Modell- und Anwendungs-Spezifikation dadurch ab, daß sie zum einen weitergehende Spezifikationen bis in den Bereich der Modell-Berechnung erlaubt und zum anderen zunehmend den Anwender anstelle des Systementwicklers als Akteur vorsieht. Die Erklärungs-Komponente schließlich kann den in Kapitel 6.2.2.2 beschriebenen Konstrukten der Ursachenanalyse und damit der Anwendungs-Spezifikation zugerechnet werden, allerdings mit erheblich größerem (flexiblerem) Leistungsumfang.

Somit kann das allgemeine konzeptionelle Modell aus Kapitel 6.1 auch auf die Konstrukt-Analyse wissensbasierter DSS-Generatoren übertragen werden. Das Konstrukt-Spektrum der Modell-Spezifikation wird dabei erweitert um Aspekte der Berechnungs-Steuerung, die nunmehr explizit als Wissensart definiert werden können. Die Anwendungs-Spezifikation betrifft konsequenterweise alle Aspekte der Auswertung der Wissensbasis, also alle Typen von Fragestellungen, die durch das wissensbasierte System beantwortet werden können. In Übereinstimmung mit der Beschreibung konventioneller DSS-Generatoren können diese Fragestellungen in allgemeine, vom Problemtyp unabhängige Aufgaben unterteilt werden, wie z.B. Zielsuche, Ursachen- und Sensitivitätsanalyse und Fallvergleiche.

Neben dieser elementaren Klassifizierung legt die Erweiterung des Spektrums der Modellspezifikation um explizit definierbare Elemente der Anwendung bei wissensbasierten DSS-Generatoren auf einer höheren Ebene die Unterscheidung nach dem Kriterium des Problemlösungstyps nahe. Diese Sichtweise der Anwendungs-Spezifikation weicht insofern von der bei konventionellen DSS-Generatoren ab, daß sie im Kern aus der Auswahl und Kombination geeigneter Konstrukte der Modellspezifikation und -berechnung (Problemlösungsstrategien) besteht. Die bereits früh formulierte idealistische Zielvorstellung, verschiedenste Problemtypen, d.h. Anwendungen mit ein und derselben Modellbasis, d.h. Wissensrepräsentation lösen zu können (vgl. [Newell 82, Knowledge Level]), ist jedoch bis heute unerreicht. Vielmehr haben sich (auch hier) auf bestimmte Problemtypen spezialisierte, wissensbasierte DSS-Generatoren entwickelt.

Für das Kriterium des Problemtyps werden in der Literatur verschiedene Klassifizierungen beschrieben (vgl. [Hayes-Roth 83, Building Expert Systems]; [Clancey 85, Heuristic Classification]; [Breuker 89, Models of Expertise]; [Chandrasekaran 87, Generic Tasks]; [McDermott 88, Problem-Solving Methods]). Einen umfassenden Vergleich gibt Puppe in [Puppe 90, Problemlösungsmethoden], S. 18-29. Angesichts des hier gesetzten Ziels der Konstrukt-Analyse ist dessen Einteilung am besten geeignet, da sie sich an der Zuordnung von Problemlösungsstrategien, im hier verwendeten Sprachgebrauch also Modellierungs-Konstrukten, zu Problemtypen orientiert. Dementsprechend werden unter Anwendungs-

Spezifikation auf einer zweiten, höheren Ebene die Anwendungsklassen *Diagnostik, Konstruktion* und *Simulation* betrachtet.

Die Darstellung der Konstrukte zur Modellspezifikation folgt der Terminologie wissensbasierter Systeme. Da vieles - vor allem im Bereich des Faktenwissen - jedoch bekannten Konstrukten konventioneller Systeme entspricht bzw. darauf aufbaut, wird jeweils der Bezug dazu hergestellt. Ferner werden die Konstrukte wieder anhand der Realisierung in konkreten Werkzeugen illustriert.

6.3.2. Modell-Spezifikation (Wissensrepräsentation)

6.3.2.1. Faktenwissen

Die einfachste Form Faktenwissen zu repräsentieren sind *Attribut-Wert-(AW)Paare*. Sie entsprechen dem gewöhnlichen (skalaren) Variablen-/Konstanten-Konstrukt, z.B. Personalkosten_Kostenstelle_1, Lohnquote_Kostenstelle_2 usw.. Da hiermit keine Strukturen abgebildet werden können, wird in wissensbasierten Systemen von einer *flachen* Wissensrepräsentation gesprochen.

Die benannte Zusammenfassung mehrerer Attribut-Wert-Paare führt zu sogenannten *Objekt-Attribut-Wert-(OAW)Tripeln*. In obigem Beispiel werden die Personalkosten und Lohnquote zu Attributen der Objekte Kostenstelle 1 bzw. Kostenstelle 2. Die Beziehungen zwischen den Elementen des Tripel weisen eine feste Bedeutung (Semantik) auf: Das Objekt *hat-ein* Attribut, das Attribut *ist-ein* Wert.

OAW-Tripel können somit als Spezialisierung eines allgemeineren Konstrukts interpretiert werden, der *semantischen Netze*, die als atomare Beschreibungselemente nur Knoten und Kanten unterscheiden: Objekte, Attribute und Werte stellen inhaltlich festgelegte Ausprägungen der Knoten, die Beziehungen Spezialfälle der Kanten dar. Da in semantischen Netzen weder Knoten noch Kanten typmäßigen Beschränkungen unterliegen, resultiert ein universell einsetzbares Instrument zur Repräsentation von Faktenwissen: Knoten können Objekte verschiedenster Abstraktionsebenen, Kanten beliebige Relationen beschreiben. Zusammengesetzte Objekte repräsentieren sich als Teilmengen des semantischen Netzes, die in ihrer Gesamtheit durch geeignete Zugriffsprozeduren, d.h. eine Traversierung des Netzes über bestimmte Kantentypen "angesprochen" werden können. Andere interpretative Prozeduren gestatten die Ableitung von Aussagen für Knoten aus abstrakteren Netzteilen, wodurch Vererbung simuliert wird. Aufgrund der statischen Grundkonzeption eignen sich semantische Netze hauptsächlich für Wissensgebiete mit einer stabilen Taxonomie (vgl.

[Kurbel 89, Expertensysteme], S. 40). Andere Autoren sehen sie mehr als graphische Darstellung objektorientierter Repräsentationsformen (vgl. [Nilsson 82, Artificial Intelligence], S. Kap. 9.2; [Puppe 91, Expertensysteme], S. 32), die konzeptionell auf semantischen Netzen aufbauen, deren Konstrukte jedoch Spezialisierungen und Erweiterungen dahingehend aufweisen, daß zusammengesetzte Objekte (Teilmengen des Netzes) mit Struktur und Verhalten im Mittelpunkt stehen.

Diese Erweiterungen führen zu einem ersten, für wissensbasierte Systeme spezifischen Konstrukt, den von Minsky 1975 zur Abbildung stereotyper Situationen eingeführten *Frames* (vgl. [Minsky 75, Frames]). Die Erweiterungen folgen der Spezialisierungs-Richtung von OAW-Tripeln und liegen in drei Bereichen:

- ❒ der Ausweitung der Objekt-internen Struktur in die Breite, indem - wie bei OAW-Tripeln - mehrere Attribute, hier Slots genannt, zusammengefaßt werden,
- ❒ der Erhöhung des Organisationsgrades durch die Einführung von Hierarchieebenen zwischen den Objekten und darauf operierenden Vererbungsmechanismen sowie
- ❒ der Ausweitung der Objekt-internen Struktur in die Tiefe, indem außer Werten weitere Informationen über Attribute, sogenannte Facetten oder Meta-Slots, gebunden werden können.

Abbildung 6-12 repräsentiert am Beispiel des Objekt-Editors im Entwicklungswerkzeug NexpertObject die ersten beiden Erweiterungen: die Property-Einträge realisieren die objektinterne Breite, die Einträge unter SuperClass und SubObject die (hierarchischen) Beziehungen zu anderen Objekten. Sie entsprechen praktisch den Kanten zwischen Teilnetzen in semantischen Netzen. Je nach Kantentyp sind zwei Effekte zu unterscheiden:

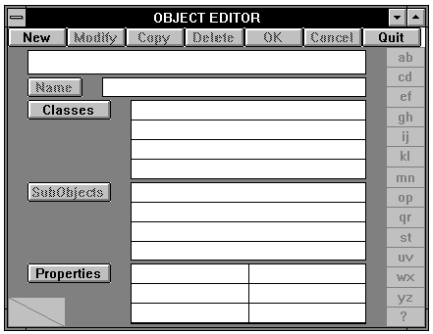


Abb. 6-12:	Objekt-Spezifikation in NexpertObject

- ❒ die Aufteilung von Attributen (Objekten) auf Teil-Objekte (part-of bzw. SubObject) und
- ❒ die Zuordnung von Objekten zu Objekt-Typen (is-a bzw. SuperClass).

Während der erste Effekt lediglich (hierarchische) Zwischenstufen der Attribut-Zusammenfassung Redundanz-neutral realisiert, basiert der zweite Fall auf einer typmäßigen Unterscheidung von Objekten in generische Objekte (Klassen) und individuelle Ausprägungen (Instanzen). Dies führt zu einer erheblichen Reduktion von Komplexität und Redundanz, indem für eine Gruppe von Objekten (Instanzen oder Klassen) allgemeingültige Fakten (und Prozeduren) in generischen (Ober-)Klassen abgelegt und durch geeignete Prozeduren "abgeleitet" werden können (Vererbung; vgl. Kapitel 6.3.2.2 und 8.1.1.2). Zusätzliche implizit generierte oder explizit spezifizierbare Attribute bzw. Slots, z.B. "AKO" (a kind of) in FRL oder SuperClass in NexpertObject (vgl. Abb. 6-12), realisieren die Vorgänger-Nachfolger-Beziehung und damit die hierarchische Struktur. NexpertObject erlaubt dabei die Definition von mehr als einem Vorgänger, was über multiple Vererbung für eine weitere Verringerung von Redundanzen und Erhöhung von Flexibilität eingesetzt werden kann.

Die Ausweitung der Tiefe der Objekt-internen Struktur von Frames besteht darin, daß an ein Attribut, hier zumeist *Slot* oder *Property* genannt, nicht nur ein Wert sondern weitere Wissenselemente, sogenannte *Facetten* oder *Meta-Slots*, gebunden werden können. Der Zweck dieser Facetten reicht von der Repräsentation von Zusatzinformationen des Slots, z.B. einer verbalen Beschreibung, über zusätzliche Eigenschaften des Werts, z.B. Datentyp und Dimensionalität, der Festlegung einer zulässigen Wertemenge oder eines Standardwertes (Erwartungswert), bis hin zu ereignis-orientierten, prozeduralen Formen der Bestimmung des Wertes bzw. daraus resultierender Konsequenzen, die in den Bereich von Ableitungs- und Kontrollwissen verweisen. Die Abb. 6-13 zeigt ein Beispiel der Slot-Spezifikation in dem Entwicklungswerkzeug KappaPC.

Als *Datentypen* für die Wert-Facette (value type) stehen neben alpha-numerisch, numerisch und logisch in NexpertObject zusätzlich Datum und Zeit, in KappaPC insbesondere Object zur Verfügung. Dadurch wird allerdings keine hierarchische Strukturbeziehung wie mit SubObject in NexpertObject erzeugt, vielmehr werden (textuelle) Einträge als Objektnamen interpretiert. Ähnliches gilt für Datum und Zeit, die lediglich exakte, quantitative und punktbasierte Angaben erlauben. Weitergehende Ansätze zur Zeitrepräsentation, die zur Verwaltung zumeist (externe) Zeitdatenbanken mit einer Abfragesprache verwenden, ermöglichen Intervalle, qualitative Angaben, einfache oder mehrfache Referenzereignisse, z.B. "vor Kapitalschnitt" sowie Zeitreihen. Beispiel-Konstrukte finden sich hauptsächlich in Werkzeugen, die auf bestimmte Problemtypen, besonders Diagnostik und Simulation, spezialisiert sind:

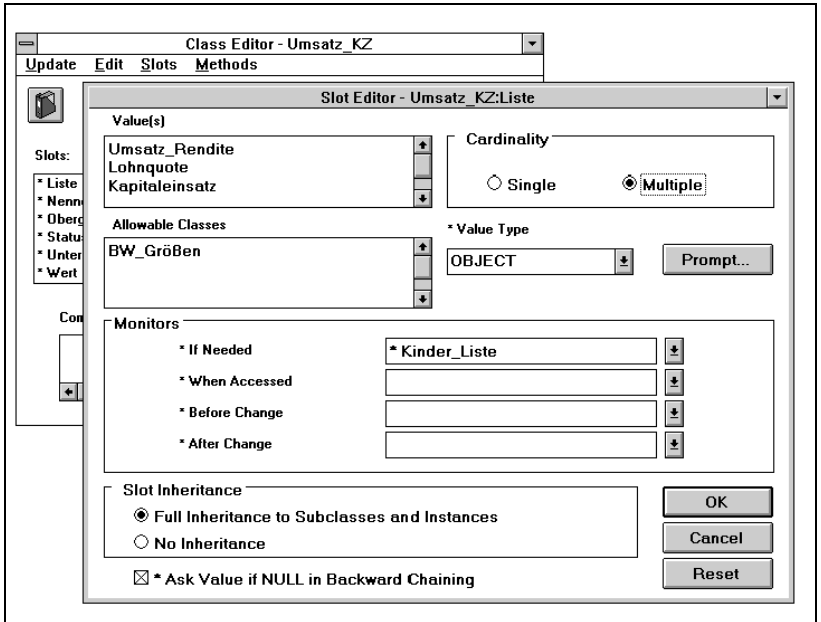


Abb. 6-13: Facetten der Slot-Spezifikation in KappaPC

VM (vgl. [Fagan 84, Monitoring Task], Kap. 22), das zur Überwachung von Patienten an der Eisernen Lunge entwickelt wurde, verwaltet Zeitreihen auf einer absoluten Zeitskala, die Diagnostik-Shell MED2 (vgl. [Puppe 87, Diagnostisches Problemlösungen], S. Kap. 4.2.10) erlaubt relative Zeitangaben zu selbstdefinierten Bezugszeitpunkten; Beispiele für ungenaue, intervallbasierte Zeitrepräsentationen sind der Temporal Map Manager (TMM) von Dean und McDermott (vgl. [Dean 87, Temporal DB-Management]) sowie der Zeitkalkül von Allen (vgl. [Allen 83, Temporal Intervals]). Ausführliche Darstellungen aller Aspekte des temporalen Schließens, insbesondere der analog zur Zeitarithmetik konventioneller Systeme entwickelten speziellen Zeitprädikate und -funktionen, finden sich bspw. in [Puppe 91, Expertensysteme], S. 65 ff. und [Charniak 85, Artificial Intelligence], S. Kap. 7.4.

Je nach Datentyp können in KappaPC *Restriktionen* für zulässige Wertebereiche spezifiziert werden, im numerischen Fall als ein Intervall aus Ober- und Untergrenze, im Fall von Objekten als Liste von Objekten oder Objektklassen. Als konkrete Anwendung lassen sich so recht einfach dynamische, datengetriebene Auswahllisten erzeugen, z.B. für dialog-orientierte Wertzuweisungen. Die Restriktionen wirken aber nur im Falle von Benutzereingaben.

Die Erweiterung der Repräsentation von Restriktionen um Variablen und Relationen (Gleichungen und Ungleichungen) führt zum Konstrukt der *Constraints*. Sie entsprechen den Nebenbedingungen konventioneller Optimierungsprobleme und beschreiben die Grenzen eines mehrdimensionalen, zulässigen Lösungsraums. Im Gegensatz zu Regeln handelt es sich um ungerichtete Beziehungen, die nach jeder Variablen aufgelöst werden können. Die Auswertung von Constraints geschieht durch spezielle Algorithmen der Problemlösungskomponente, die sich im wesentlichen als Weitergabe beschränkter Wertemengen von Variablen an direkt oder indirekt von diesen abhängige Variablen gestaltet (Constraint-Propagierung). Die Leistungsfähigkeit hängt entscheidend davon ab, ob nur feste Werte, Wertemengen (Intervalle) oder Symbole im Sinne einer (Un-)Gleichungstransformation propagiert werden können. Die Umsetzung beschränkt sich zumeist auf spezialisierte Anwendungen, z.B. die Simulation elektrischer Schaltkreise (vgl. [Stallman 77, Circuit Analysis]) oder allgemeine Constraintsprachen (vgl. [Sussman 80, Constraints]; [Hentenryck 89, Constraint Satisfaction]; [Güsgen 89, Constraint Satisfaction]).

Neben Restriktionen kann häufig die *Kardinalität* des Werte-Slot festgelegt werden, entweder einstellig (single) oder als Liste (multiple). Weitergehende Werkzeuge gestatten auch eine exakte Dimensionierung; diese bleibt aber in allen Fällen ein-dimensional, so daß das mehr-dimensionale Konstrukt konventioneller Systeme (s. Kapitel 6.2.1.1) in keinem Fall erreicht wird. Somit lassen sich typische betriebswirtschaftliche Strukturen, z.B. Kostenarten je Produkt und Zeit nur aufwendig oder in starrer, hierarchischer Struktur (Sicht) repräsentieren. Daraus resultiert das Charakteristikum einer eher punktuellen, iterativen denn globalen, simultanen Abbildung von Problemstellungen.

Abb. 6-14 zeigt einen Ausschnitt eines Anwendungsbeispiels der beschriebenen Konstrukte für die Objekt-/Frame-orientierte Abbildung von betriebswirtschaftlichen Kennzahlen, die als Basis für eine wissensbasierte Analysekomponente in einem Führungsinformationssystem dienen kann.

Hierarchisierung und Vererbung werden hier bspw. wiefolgt eingesetzt: Allen Objekten gemeinsame Attribute, z.B. der einstellige numerische Wert-Slot oder ein textueller Beschreibungs-Slot, sind in der Klasse Kennzahlen definiert. Bedarfsweise Wert-Berechnungen unterscheiden sich als individuelle if-needed-Methoden auf der Ebene der Klassen Quotienten, Summen und Differenzen, die hierzu auf individuelle Slots vom Typ Objekt zurückgreifen, z.B. Zähler- und Nenner-Slot bei der Klasse Quotienten. In den Unterklassen Umsatz_Kennzahlen wird der Nenner-Slot mit dem Objekt(namen) Umsatz vorbesetzt. Die eigentlichen Instanzen, erkennbar durch gestrichelte Verbindungslinien, z.B. Umsatz-Rendite, spezifizieren dann lediglich noch den Zähler-Slot, z.B. durch den Objekt(namen)

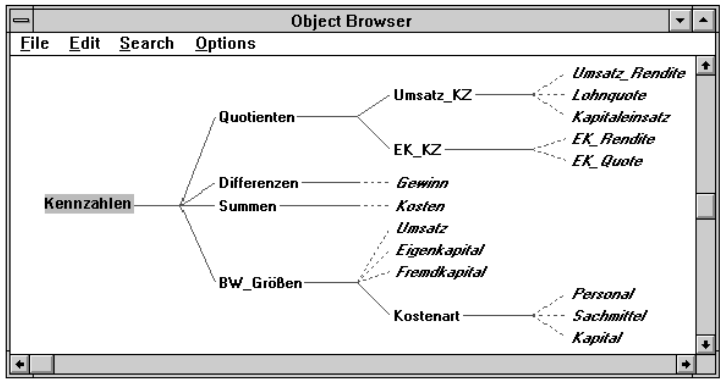


Abb. 6-14: Beispiel der Objekt-/Vererbungs-Hierarchie in KappaPC

Gewinn; alternativ kann auch hier eine vererbte, generelle if-needed-Prozedur, z.B. Objekt_Wahl in der Klasse Kennzahlen, realisiert werden. Die Abfrage des Wert-Slots von Umsatz_Rendite löst dann rekursiv die Berechnung der Differenzen-Instanz Gewinn sowie der Summierung der Kostenart-Instanzen in den Wert-Slot der Summen-Instanz Kosten, evtl. sogar die Abfrage des Zähler-Objekts aus der Slot-Option erlaubter Werte, aus.

6.3.2.2. Ableitungswissen

Als weitere Typ-mäßige Erweiterung der Objekt-internen Struktur treten prozedurale Elemente hinzu, die angeben, unter welchen Bedingungen (Ereignissen) konkrete Slots (Werte) wie zu erschließen (berechnen) sind. Damit überschreiten Frames (Objekte) die Grenze deklarativer Wissensrepräsentation, die auf die Darstellung konkreter Werte beschränkt ist. Sie werden hier als Spezialfall von Ableitungswissen aufgefaßt, bei dem die Voraussetzungen für die "Ableitung" und der Anstoß zur Anwendung determiniert sind.

Typische Beispiele für dieses als *zugeordnete Prozeduren* (attached procedures) bezeichnete Konstrukt sind die Facetten $if-added, $if-removed und $if-needed in der Frame Representation Language (FRL) von Roberts (vgl. [Roberts 77, FRL]). In objekt-orientierten Werkzeugen, wie KappaPC und NexpertObject, wird der allgemeinere Begriff der *Methoden* verwandt; auch tendieren die implementierten Konstrukte bezüglich der auslösenden Ereignisse

in eine allgemeinere Richtung. So decken die Konstrukte von KappaPC bspw. folgende Fälle ab (vgl. Abb. 6-13):

- *if-needed-method*
  wird aktiviert, wenn ein noch undefinierter Slot abgefragt oder in einer anderen Berechnung benötigt wird
- *before-change-method*
  wird aktiviert, bevor dem Slot ein (neuer) Wert zugewiesen wird, z.B. für bestätigende oder korrigierende Rückfragen beim Benutzer bzw. weitergehende Konsultationen der Wissensbasis.
- *after-change-method*
  wird aktiviert, nachdem dem Slot ein (neuer) Wert zugewiesen wird, z.B. zur Abbildung von Folgewirkungen; als Spezialfall ist die o.g. if-removed-Funktionalität enthalten.

Before-change- und after-change-method stellen eine Differenzierung der $if-added-Prozedur von FRL dar. NexpertObject sieht nur das if-change-Konstrukt vor.

Auch in konventionellen Systemen lassen sich Analogien zu den bisher beschriebenen Konstrukten finden. Beispiele für deklarative Facetten sind Memo-Felder in Planungssprachen wie CA-Compete oder Datenbanksystemen. Letztere, z.B. NOMAD, haben unter dem Stichwort der referentiellen Integrität prozedurale Konstrukte entwickelt, die beim Einfügen oder Löschen von Datensätzen automatisch aktiviert werden. Desweiteren korrespondieren virtuelle, berechnete Datenfelder oder -tabellen (virtual sets) mit if-needed-Prozeduren oder LOOKUP-Konstrukte mit Facetten zur Beschränkung des Wertebereichs, z.B. $require von FRL.

Die gängigste und populärste Form zur Repräsentation von Ableitungswissen ist das Regel-Konstrukt. Dies läßt sich damit begründen, daß es Experten in der Regel leichter fällt, ihr Wissen in regelbasierter Form zu beschreiben. Regeln bestehen grundsätzlich aus einer Vorbedingung (Prämisse) und einem Aktionsteil (Conclusio). Die Vorbedingung beschreibt die strukturelle und wertmäßige Konstellation der Fakten- oder Datenbasis, unter der der Aktionsteil ausgeführt werden darf.

Zur Formulierung der Vorbedingung stehen üblicherweise Vergleichoperatoren wie <,>,=,<>, LIKE, CONTAINS usw. sowie Verknüpfungsoperatoren wie AND, OR, NOT usw. zur Verfügung. Um Vorbedingungen und Aktionen auf dieser Basis nicht (frei) "programmieren" zu müssen, bieten Entwicklungswerkzeuge wie z.B. NexpertObject oder HEXE vorgefertigte Konstrukte (Prädikate) an, die im Extremfall kontext-sensitiv eine

Auswahl der Parameter aus der Faktenbasis erlauben. Diese Auswahl- und Konfigurationshilfen entsprechen den in konventionellen Datenbanksystemen oder EIS-Generatoren z.B. zur Filterdefinition oder Datenselektion bekannten Auswahlmenüs (vgl. Kapitel 5.3.2, Abb. 5-7).

Für die Gestaltung des Aktionsteils steht das gesamte prozedurale Spektrum von einfachen Wertzuweisungen bis hin zu komplexen Prozeduren und Dialogkomponenten zur Verfügung. Auch hier sind in der Regel mehrere Aktionen durch UND-Verknüpfung erlaubt, allerdings ohne automatische Konsistenzprüfung. Puppe unterscheidet dabei nach der Wirkung auf die Faktenbasis in

- *Implikationen*, die den Wahrheitsgehalt einer (neuen) Feststellung herleiten, und
- *Handlungen*, die den Zustand einer (alten) Faktenbasis verändern (vgl. [Puppe 91, Expertensysteme], S. 21).

Entsprechend differenzierende Konstrukte finden sich nur näherungsweise in Entwicklungswerkzeugen wieder: so unterscheidet NexpertObject zwar zwischen (genau einer) *Hypothese* je Regel, die wahr, falsch, unbekannt oder nicht erschlossen sein kann, und *Seiteneffekten* für beliebig viele, weitere Aktionen, dies dient aber primär der Steuerung der Problemlösungskomponente. Abb. 6-15 zeigt ein Beispiel einer Ableitungsregel für eine Erweiterung des Modells aus Abb. 6-14.

Da sich Regeln in konkreten Anwendungen häufig strukturell stark ähneln, d.h. lediglich in Vergleichsobjekt und -wert unterscheiden, führt der Einsatz von Variablen in Vorbedingung und Aktionsteil zu einer erheblichen Reduktion der (zu wartenden) Regelmenge. Voraussetzung dafür ist, daß eine Teilmenge von Objekten der Faktenbasis spezifiziert werden kann. In Werkzeugen mit objekt-/frame-basierter Faktenbasis knüpft diese Teilmengenbildung an der Objekthierarchie an: als "Bedingung" sind (Sub-)Klassen anzugeben, für deren Unterobjekte die Regel gelten soll. Bspw. genügt in Abbildung 6-15 eine einzige Regel "Kritische_Kennzahl", um alle Unterobjekte der Klasse Kennzahlen daraufhin zu analysieren, ob eine Grenzwertverletzung vorliegt. Die Vergleichswerte sind dabei als Slots (Obergrenze, Untergrenze) unter Ausnutzung des Vererbungsmechanismus redundanzfrei hinterlegt.

Da das Pattern-Konstrukt in KappaPC nicht nach dem Objekttyp (Instanz oder Klasse) differenziert, müßte im Beispiel ein Ausschluß der Unterklasse Umsatz_K(enn)Z(ahlen) explizit im Bedingungsteil aufgenommen werden. Das Pattern-Konstrukt von NexpertObject überspringt dagegen grundsätzlich alle Unterklassen bis zur obersten Instanzen-Ebene.

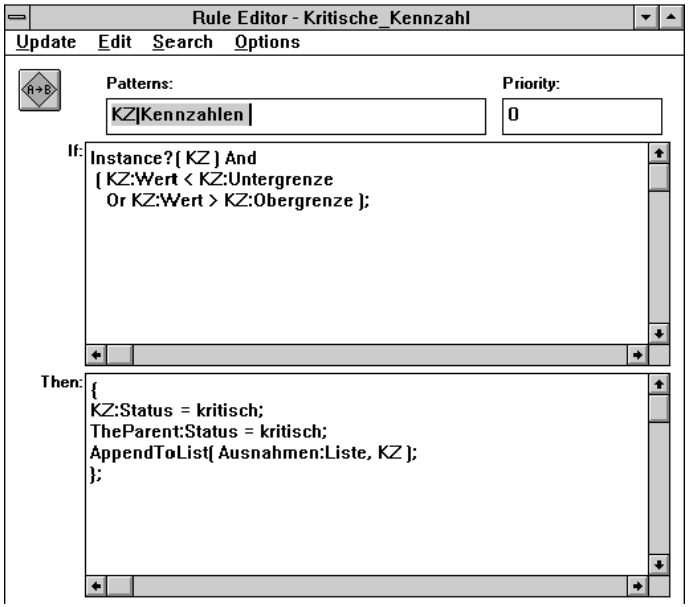


Abb. 6-15: Beispiel einer Regel-Spezifikation mit Variablen in KappaPC

Eine alternative Form der Wertbestimmung basiert auf der hierarchischen Anordnung von Objekten und damit implizit verbundenen oder darauf operierenden expliziten Vererbungsmechanismen.

Die Wertbestimmung durch *Vererbung* kann als implizite if-needed-Prozedur interpretiert werden, die als Parameter den AKO-Slot rekursiv auswertet. In KappaPC sind ererbte Werte oder Methoden durch einen Stern (*) gekennzeichnet (vgl. Abb. 6-12). Weitergehende Implementierungen dieses Konstrukts gestatten Parametrisierungen der Vererbungsprozedur, die über weitere, spezifische Facetten definiert werden. Abb. 6-16 zeigt am Beispiel von NexpertObject Ausprägungen dieser dort als Meta-Slots bezeichneten Wissenselemente.

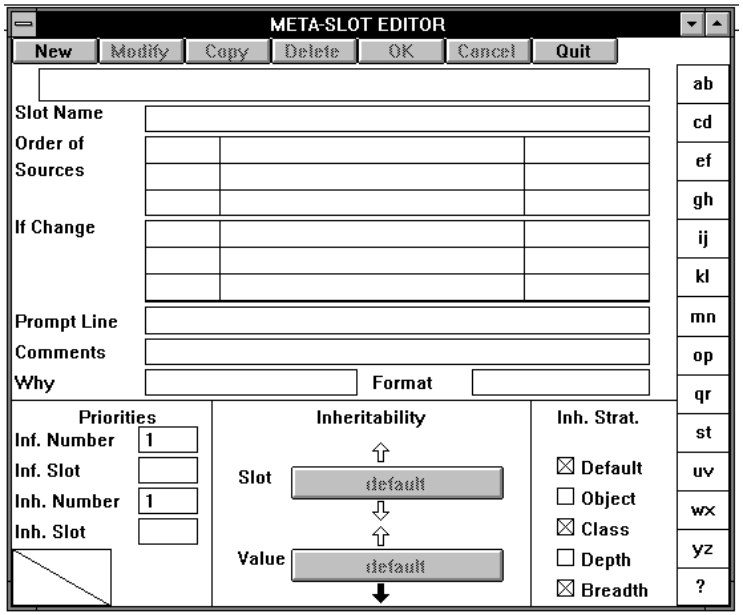


Abb. 6-16: Meta-Slots in NexpertObject

Dabei wird zwischen grundsätzlicher Vererbbarkeit (inheritability) und deren Form (inheritance strategy) unterschieden. Die Vererbbarkeit kann getrennt für die Eigenschaft (slot) und deren Wert (value) angegeben werden. Andere Werkzeuge realisieren dies durch Slottypen, z.B. vererbbare *member-slots* und nicht vererbbare *own-slots* in KEE. In NexpertObject können Werte neben der üblichen top-down-Richtung vom Generellen zum Speziellen auch umgekehrt "vererbt" werden. Dadurch lassen sich verallgemeinernde Schlüsse leicht abbilden, z.B. wenn eine konkrete Kennzahl kritisch ist, dann soll auch die Gesamtheit aller Kennzahlen als kritisch eingestuft werden. Dadurch können in anderen Werkzeugen dafür notwendige Regeln eingespart werden.

Durch Mehrfach- und Aufwärts-Vererbung können grundsätzlich Konflikte bei der Wertzuweisung auftreten. Zur Auflösung dieser Konflikte können weitere Meta-Slots spezifiziert werden: Inheritance-Category gestattet die Auswahl nach (numerischen) Prioritäten, entweder direkt (Inference-Number) oder als Verweis auf einen beliebigen (numerischen) Slot auch anderer Objekte (Inference-Slot); Inheritance-Strategy bestimmt Typ und Richtung der Auswahl, wobei zwischen Objekt (Instanz) oder Klasse und zwischen Breiten- oder Tiefen-Suche gewählt werden kann. Aufgrund ihres steuernden Charakters weisen diese Objekteigenschaften bereits in den Bereich des Kontrollwissens hinein, ähnlich wie Vererbung und zugeordnete Prozeduren als Spezialfälle von Ableitungswissen aufgefaßt wurden.

6.3.2.3. Kontrollwissen

Die explizite Darstellung von Kontrollwissen ist von erheblicher Bedeutung für die Effizienz der Problemlösungskomponente (vgl. [Puppe 91, Expertensysteme], S. 28; [Kurbel 89, Expertensysteme], S. 53). Fehlen spezifische Konstrukte, muß das Kontrollwissen unter Zuhilfenahme der Repräsentationsformen für Fakten und Ableitungen vermischt abgebildet werden, z.B. in Form selbstdefinierter Attribut-Wert-Paare oder Slots, zusätzlicher Regeln bzw. Vorbedingungen in diesen oder prozeduraler Steuerblöcke für den Problemlösungsteil.

Kontrollwissen steuert die Auswahl und Reihenfolge von Fakten- und Ableitungswissen in konkreten Anwendungssituationen. Es ergänzt die grundsätzlichen Kontrollstrukturen, die fest in der Problemlösungskomponente implementiert sind und die in Kapitel 6.3.4 behandelt werden. Dabei können zwei grundsätzliche Formen unterschieden werden:

- ❒ (faktische) Parameter, die von den Inferenz- und Konfliktlösungs-Komponenten interpretiert werden und
- ❒ (prozedurale) Anweisungsblöcke, die die Problemlösungskomponente ergänzen.

Die im vorangegangenen Kapitel beschriebenen Parameter Inheritance-Category/-Slot und Inheritance-Strategy bei Vererbungsprozessen sind ein Beispiel für die erste Gruppe. Zur Behandlung analoger Konfliktfälle bei Regeln sind ähnliche Konstrukte üblich, z.B. Rule-Priority in KappaPC (vgl. Abb. 6-15) oder (Rule) Inference Category bei NexpertObject. Im letzten Fall stehen wieder zwei alternative Konstrukte zur Auswahl, die direkte numerische Angabe als (Rule) Inference-Priority-Number oder die indirekte Ableitung aus einem (numerischen) Slot als (Rule) Inference-Priority-Slot (vgl. Abb. 6-17).

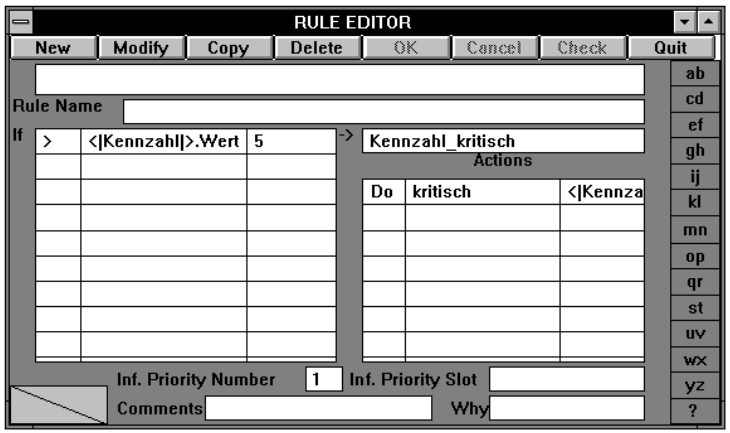


Abb. 6-17: Regel-Spezifikation in NexpertObject

Diese Form der Priorisierung betrachtet Regeln als Ganzes, während der Bedingungsteil implizit Detailinformationen über deren Anwendbarkeit enthält. Durch Gewichtung der in Vorbedingungen verwendeten Fakten kann eine differenziertere Auswahl getroffen werden. Dieses Prinzip ist in NexpertObject durch das Meta-Slot-Konstrukt Inference-Category, wieder alternativ als Wert (Inference-Number) oder Slot-Verweis (Inference-Slot), realisiert (vgl. Abb. 6-16). Dabei wird diejenige Regel bevorzugt, die das Faktum mit der größten Priorität verwendet; zusammengesetzte Bewertungen, z.B. Summe oder Durchschnitt aller Fakten-Prioritäten in den Regel-Vorbedingungen, sind aber nicht möglich.

Außer der Reihenfolge kann Kontrollwissen auch Einfluß darauf nehmen, ob und wie eine Regel angewendet wird. Bei NexpertObject wird dies bspw. durch mit Wirkungen vorbelegte Intervalle der Regel-Prioritätszahl (Inference-Category) realisiert: z.B. sperren Werte kleiner -20 die Regel, selbst wenn alle Bedingungen erfüllt sind, oder unterbinden Werte zwischen -5 und -1 alle Seiteneffekte der Regel, so daß nur (boolesche) Hypothesen abgeleitet werden können. Dies kann in Führungsinformationssystemen bspw. dazu eingesetzt werden, in Abhängigkeit vom Ausmaß einer Abweichung Detailanalysen anzustoßen oder zu unterlassen.

Regeln und Vererbung sind nur zwei Möglichkeiten zur Ermittlung von Fakten. Als weitere Alternativen kommen direkte Benutzereingabe, Datenbankabfrage oder Berechnung durch sonstige, beliebige andere Anwendungssysteme in Betracht. Zur Steuerung der Reihenfolge dieser Alternativen wurden spezielle Konstrukte entwickelt, z.B. die Sources-Facette in ESE oder der Meta-Slot OrderOfSources in NexpertObject (vgl. Abb. 6-16). Dabei handelt es sich praktisch um Prozeduren, die zum einen Standards der Problemlösungskomponente Faktum-spezifisch überschreiben, und die zum anderen eine Realisierung in Form von zusätzlichen Regeln ersparen.

Eine andere Form der expliziten Repräsentation von Kontrollwissen basiert auf der Strukturierung der Regelmenge. Im einfachsten Fall werden Regeln zu Gruppen zusammengefaßt, die dann als explizite Vorauswahl der Schlußfolgerungskomponente zur Auswertung übertragen werden. Dies kann direkt und statisch durch Aufzählung von Regeln geschehen, wie z.B. bei *RuleGroups* in ESE oder *RuleSets* in KappaPC, oder indirekt und dynamisch durch das *Kontext*-Konstrukt, wie z.B. in NexpertObject.

Im letzten Fall werden Hypothesen (boolesche Variablen oder Objekt-Slots) miteinander verknüpft, deren diese erschließenden Regelmengen mangels Regelverknüpfung nicht direkt von der Schlußfolgerungskomponente in Zusammenhang gebracht werden können (knowledge islands). Dadurch kann der Ableitungsprozess dynamisch auf die Bestimmung von Fakten ausgeweitet werden, ohne diese explizit in Auftrag geben zu müssen, bspw. die Analyse der Kapazitätsauslastung in Abhängigkeit von Plan-Ist-Abweichungen des Umsatzes.

Komplexere Formen der Regelstrukturierung basieren auf der Einführung von Regelhierarchien, speziell in Form von übergeordneten Steuerungs- oder Meta-Regeln, wie z.B. in PersonalConsultantPlus (PCPlus). Meta-Regeln können der Ermittlung einzelner Fakten, in PCPlus Parameter genannt, oder einer Gruppe von Fakten, in PCPlus Teilproblem bzw. Frame genannt, zugeordnet werden. Ihre Ausführung ist dem eigentlichen, normalen Schlußfolgerungsprozeß vorgelagert und dient im wesentlichen der Veränderung der Wertigkeit und Reihenfolge von Regeln. Da auch die Erschließung anderer Teilprobleme angestoßen werden kann, die ihrerseits über Meta-Regeln verfügen können, lassen sich so beliebig tief in Teilprobleme dekomponierte und dynamisch gesteuerte Problemlösungen realisieren. Bspw. können Meta-Regeln darüber entscheiden, ob bzw. wann eine Liquiditätsanalyse durchgeführt wird.

Ähnliche Effekte erzielt das Konstrukt der Focus-Control-Blocks (FCB) in ESE. Da jedoch Meta-Regeln fehlen, reduziert sich die Steuerung auf rein prozedurale Anweisungsfolgen

innerhalb der FCB's sowie "Steuerungs-Aktionen" im Schlußfolgerungsteil normaler Inferenzregeln. Ebenso werden dynamische Veränderungen der Kontroll-Parameter in Nexpert-Object durch Seiteneffekte normaler Inferenzregeln abgebildet.

Neben den beschriebenen expliziten, konstruktmäßigen oder impliziten Formen der Abbildung von Kontrollwissen in der Wissensbasis kann der Problemlösungsprozeß über globale Parameter und Strategien beeinflußt werden. Diese werden im Kapitel 6.3.3.3 näher dargestellt (vgl. Abb. 6-18).

6.3.3. Modell-Berechnung

6.3.3.1. Funktionale Aspekte

Die Modell-Berechnung, verstanden als die Ermittlung nicht-originärer, d.h. nicht explizit eingegebener Werte aus originären oder zuvor berechneten Werten, wird in wissensbasierten Systemen von der *Problemlösungskomponente* wahrgenommen. Da die Werte mit Hilfe von Ableitungs- und Kontrollwissen aus Faktenwissen *abgeleitet* werden und eher qualitativen denn quantitativen Charakters sind, wird zumeist nicht von *Berechnen* sondern von (Er-) *Schließen* bzw. der *Inferenzkomponente* gesprochen. Dabei sind zwei Hauptfunktionen zu unterscheiden:

- Die *Inferenz*, das Schließen im engeren Sinne:
  Darunter wird die Realisierung einzelner Schritte des "Wissenszuwachses" in Bezug auf eine konkrete Problemstellung verstanden. Theoretisch basiert die Legitimation des Wissenszuwachs auf übergeordneten Schlußfolgerungskalkülen, z.B. dem modus ponens oder der Resolution (vgl. [Kurbel 89, Expertensysteme], S. 8; [Puppe 91, Expertensysteme], S. 18). Mechanistisch betrachtet geschieht dies durch die Auswertung einzelner Elemente des Ableitungswissens.

  Jede dieser Aktionen verändert unmittelbar Umfang und Inhalt der Ausgangs-Wissensbasis und repräsentiert den "Wissenszuwachs" im Kontext einer konkreten Problemstellung. Der Wissenszuwachs ist dabei nicht nur auf Fakten beschränkt, sondern schließt Ableitungs- und Kontrollwissen ein, z.B. wenn Prioritäten von Fakten oder Regeln verändert werden.

- Die *Kontrolle*, das Steuern und Überwachen der Inferenz:
  Dies umfaßt die zielgerichtete Auswahl und Sequentialisierung auszuwertender Elemente des Ableitungswissens. Hierzu gehören Startpunkt und Richtung der Lösungssuche sowie Behandlung von Konfliktsituationen. Zur Steuerung gehört auch die Überwachung der Gültigkeit des erschlossenen Wissenzuwachses bis hin zur (teilweisen) Revision, dem nicht-monotonen Schließen.

6.3.3.2. Inferenzstrategien

Im einfachsten Fall bedeutet dies die Ausführung des Aktionsteils einer Regel, einer zugeordneten Prozedur eines Frame oder die Auswertung von Vererbungspfaden. Daneben sind komplexere Fälle zu unterscheiden, bei denen Fakten der Wissensbasis durch einen spezifischen Algorithmus (Konstrukt) der Problemlösungskomponente ausgewertet werden:

Ein für Führungsinformationssysteme bedeutsames Beispiel ist die Verarbeitung unsicheren Wissens durch *probabilistisches Schließen*. Dabei werden alle Aussagen (Fakten, Implikationen, usw.) mit Wahrscheinlichkeitswerten belegt, die durch Expertenschätzung oder statistische Auswertungen, z.B. auf der Basis von Falldatenbanken, gewonnen werden. Verrechnungsalgorithmen sorgen dann für die problemspezifische Propagierung der (Ausgangs) Wahrscheinlichkeiten über die Wissensbasis. Exakte statistische Verfahren basieren auf einer Vielzahl praktisch nicht realisierbarer Voraussetzungen: so erfordert die Anwendung des Bayes-Theorems zur Errechnung der (bedingten) Wahrscheinlichkeit einer Schlußfolgerung aus den bedingten Wahrscheinlichkeiten ihrer Vorbedingungen und der unbedingten Wahrscheinlichkeit der Schlußfolgerung (a-priori-Wahrscheinlichkeit), daß sowohl die Vorbedingungen untereinander unabhängig als auch im Bezug zur Schlußfolgerung eindeutig sind sowie die Schlußfolgerungen sich gegenseitig ausschließen. Neben Ansätzen methodischer Verfeinerung, z.B. der Dempster-Shafer-Theorie (vgl. [Gordon 85, Evidential Reasoning]), werden zumeist vereinfachte (heuristische) Verrechnungsschemata mit erweiterten, qualitativen Unsicherheitsmaßen eingesetzt. So verwendet das Expertensystem INTERNIST zur Diagnose innerer Erkrankungen (vgl. [Miller 82, INTERNIST]) eine dreidimensionale Punkte-Bewertung von Symptomen (Vorbedingungen):

- Die *Evoking-Strength* drückt aus, wie stark ein Symptom für eine Diagnose spricht,
- die *Frequency-Kennzahl* gibt an, wie stark ein Symptom bei einer Diagnose zu erwarten ist und
- der *Import-Value* gibt umgekehrt an, wie stark das Nichtvorhandensein eines Symptoms gegen die Diagnose spricht.

Die Unsicherheitsindikatoren werden vom Experten auf einer sechsstufigen Skala qualitativer Bewertungen geschätzt und durch ein zugeordnetes Plus-Minus-Punkteschema verrechnet. Ein Vorteil liegt also in der getrennten Bewertung positiver und negativer Evidenz,

dafür werden weder evtl. synergetische Kombinationen von Vorbedingungen (Symptomen) noch Apriori-Wahrscheinlichkeiten berücksichtigt.

Ein einfacher Ansatz zur Berücksichtigung von Symptom-Kombinationen, der sich auch häufig in allgemeinen Entwicklungswerkzeugen wie z.B. PersonalConsultantPlus oder ESE findet, geht auf das Expertensystem MYCIN zurück (vgl. [Shortliffe 75, Inexact Reasoning]). Die Gesamtwahrscheinlichkeit einer komplexen Vorbedingung mit UND- bzw. ODER-Verknüpfungen wird dabei als Maximum bzw. Minimum der Einzelwahrscheinlichkeiten der Vorbedingungen ermittelt. Verteilungsaspekte bleiben allerdings unberücksichtigt, was zwangsläufig Gleichbehandlungen unterschiedlicher Situationen bewirkt, z.B. ist es ohne Bedeutung, ob nur eine oder alle Vorbedingungen "unsicher" sind. Die weitere Propagierung der Wahrscheinlichkeiten vollzieht sich in zwei Stufen. Zunächst wird die "apriori"-Wahrscheinlichkeit der Regel, d.h. die Expertenschätzung für das Vorliegen der Schlußfolgerung unter der Voraussetzung hundert-prozentig erfüllter Vorbedingungen, "korrigiert", indem sie mit der "Gesamtwahrscheinlichkeit" der Vorbedingung multipliziert wird. Die so gewonnenen Sicherheitsmaße für eine Diagnose (Regelwahrscheinlichkeiten) werden dann unter der Annahme der Unabhängigkeit der einzelnen Regeln additiv zu einem Sicherheitsfaktor verknüpft.

In Analogie zur Ermittlung von Gesamtwahrscheinlichkeiten verknüpfter Vorbedingungen verwendet das Expertensystem CASNET zur Glaukomdiagnostik (vgl. [Kulikowski 82, CASNET/EXPERT]) und das daraus entwickelte Entwicklungswerkzeug für Diagnose-Anwendungen EXPERT statt der Summe das Maximum der Wahrscheinlichkeiten aller auf eine Diagnose weisenden Regeln. Ein differenzierter Ansatz liegt in der Diagnostik-Shell MED1 vor (vgl. [Puppe 83, MED1]), bei dem die Summe der Evidenzen aller positiven und negativen gefeuerten Regeln in einem weiteren Schritt in eine von sieben qualitativen Wahrscheinlichkeitsklassen transformiert wird. Die Gestalt der Transformationsfunktion, z.B. progressiv, degressiv oder S-förmig, repräsentiert dabei näherungsweise die Form der Evidenzverstärkung und damit indirekt, wenn auch nur summarisch, die Korrelationen zwischen den einzelnen Regeln.

Die aufgeführten Beispiele zeigen, daß Implementierungen des probabilistischen Schließens in hohem Maße von den Eigenschaften der Wissensbasis abhängen. Daher sind sie auch weniger in allgemeingültigen denn in problemtyp-spezifischen Entwicklungswerkzeugen zu finden. Doch auch bei größter Anpassung bleibt das Grundproblem der nur bei abgeschlossenen technischen Systemen objektivierbaren Quantifizierung von Wahrscheinlichkeiten. Dem Rückzug auf wenige qualitative Bewertungen wird vorgeworfen, daß er zwar das Risiko verringere, Fehlinterpretationen scheinbar genauer Verrechnungsschemata zu erliegen,

aber nicht verhindern könne, daß verschiedene Experten zu unterschiedlichen Ergebnissen bzw. Einschätzungen gelangen. Eine solche Nivellierung real existierender Meinungsverschiedenheiten kann aber nicht das Ziel entscheidungs-unterstützender Werkzeuge sein. Vielmehr muß die Offenlegung und Nachvollziehbarkeit der Ursachen im Zusammenhang mit der Erklärungskomponente im Vordergrund stehen. Dies setzt ein höchstmögliches Maß expliziter Repräsentation von Unsicherheiten voraus. Diesem Gedanken folgen alternative Ansätze, die entsprechend der bereits 1978 von Szolovits et al. geäußerten Empfehlung weitestgehend auf Wahrscheinlichkeiten und Verrechnungsschemata verzichten (vgl. [Szolovits 78, Categorical and Probabilistic Reasoning]). Hierzu zählen die Repräsentation von Unsicherheiten als Ausnahmen von Regeln, die zur Rücknahme von Schlußfolgerungen führen (vgl. nicht-monotones Schließen in Kapitel 6.3.3.3) oder als kausale Ursache-Wirkungs-Modelle, z.B. im Rahmen der modellbasierten Diagnostik. Für eine detailliertere Betrachtung wird auf [Puppe 91, Expertensysteme], S. 43 ff. verwiesen.

6.3.3.3. Kontrollstrategien

Kontroll- oder Abarbeitungsstrategien repräsentieren das eigentliche Problemlösungsverhalten, d.h. die Auswahl und Reihenfolge des anzuwendenden Fakten- und Ableitungswissens in Bezug auf eine zu beantwortende Fragestellung. Sie sind sowohl explizit in Form von Kontrollwissen als auch implizit in Form von Prozeduren innerhalb der Problemlösungskomponente realisiert, die von Modell- oder Anwendungsspezifikationen angestoßen und durch das Kontrollwissen parametrisch beeinflußt werden. Zu den internen Prozeduren gehören insbesondere Algorithmen zur Verkettung von Regeln. Nach dem Ausgangspunkt der Abarbeitung werden *Vorwärts*- und *Rückwärtsverkettung* unterschieden.

Das *forward-chaining* oder datengetriebene Vorgehen, bei dem mögliche Schlußfolgerungen aufgrund vorzugebender Fakten gezogen werden, eignet sich besonders für Fragestellungen, die auf die Konsequenzen bekannter Tatbestände gerichtet sind. Das *backward-chaining* oder zielgetriebene Vorgehen, bei dem für ein zu bestimmendes Faktum oder eine zu beweisende Hypothese nach einer gültigen Schlußfolgerungskette gesucht wird, tendiert dagegen eher zu Fragestellungen nach den Einflußgrößen oder Ursachen bekannter oder vermuteter Zustände. Üblicherweise werden beide Grundstrategien in direktem Zusammenhang mit Regeln dargestellt (und implementiert). Wie bereits in Kapitel 6.3.2.2 angedeutet finden sich die Prinzipien aber auch implizit bei der Auswertung von Vererbungshierarchien und zugeordneten Prozeduren frame- bzw. objektorientierter Systeme: if-needed-Prozeduren realisieren eine Rückwärts-, if-changed-Prozeduren eine Vorwärtsverkettung. Vererbung ist normalerweise auf (passive) Rückwärtsverkettung beschränkt, kann aber wie im Falle von NexpertObject durch einen Kontrollparameter auch eine (aktive) Vorwärtsverkettung auslö-

sen, so daß Slotwerte auf Vorfahren oder Nachkommen übertragen (propagiert) werden. Häufig ist eine Kombination beider Vorgehensweisen sinnvoll, z.B. um einerseits zu überprüfen, ob und wo kritische Plan-Ist-Abweichungen vorliegen, und andererseits welche Konsequenzen dies für (aggregierte) Rendite-Kennziffern hat. Diese (sekundären) Konsultationsaufträge können direkt durch übergeordnete Bearbeitungsprozeduren (s. Kapitel 6.3.4. Anwendungsspezifikation) oder indirekt durch Seiteneffekte im "Schlußfolgerungs"-Teil von Regeln oder zugeordnete Prozeduren angestoßen werden.

Typischerweise treten in allen Fällen der Lösungssuche Konflikte dergestalt auf, daß mehrere Alternativen untersucht werden können, sei es daß mehrere Regeln anwendbar sind oder bei multipler Vererbung mehrere Vorgänger/Nachfolger in Frage kommen. Mechanismen zur Regelung dieser Situation werden als Konfliktlösungsstrategien bezeichnet. Aufgrund ihrer übergeordneten Aufgabe der Zielbestimmung weisen sie zugleich den Charakter von Suchstrategien auf. Einfache (dumme) Strategien "suchen" ohne Bewertungskriterium: Beispiele sind die *Tiefensuche* (depth-first), bei der die Verkettung von Regeln im Vordergrund steht, und die *Breitensuche* (breadth-first), bei der die Auswertung aller Alternativen im Vordergrund steht. Bewertungsfunktionen beeinflussen die Auswahl der jeweils als nächstes betrachteten Alternative. Im einfachsten Fall orientiert sie sich an lokalen Kriterien, z.B. der Anzahl oder dem Gewicht bereits bekannter Vorbedingungen. Dies kann in einer weiteren Stufe mit einem kontextabhängigen Wechsel des Voranschreitens in die Tiefe oder Breite kombiniert werden. Globale Bewertungsfunktionen orientieren sich an einem Schätzwert bezüglich der Zielentfernung, was allerdings Zusatzwissen erfordert.

Beispiele sind *Hill-climbing*-Verfahren, bei denen die jeweils beste Alternative in die Tiefe verfolgt wird, die *Strahlsuche*, bei der Hill-climbing auf die n besten Alternativen jeder (Breiten-)Ebene ausgeweitet wird und die *Bestensuche*, bei der jeweils die Alternative mit der größten Zielnähe in die Tiefe weiterverfolgt wird, egal auf welcher Ebene sie sich befindet. Eine umfassende Darstellung unter Einbezug optimierender Suchverfahren findet sich bspw. in [Winston 87, Künstliche Intelligenz], S. 105 ff..

Normalerweise terminiert die Problemlösungskomponente bei Rückwärtsverkettung, wenn die erste Lösung für das Ziel gefunden ist, bei Vorwärtsverkettung, wenn keine weiteren Schlüsse mehr aus den vorgegebenen Fakten gezogen werden können oder ebenfalls ein explizites Abbruchkriterium erfüllt wird. Spezielle Kontrollparameter gestatten darüberhinaus in einigen Entwicklungswerkzeugen die Ausweitung des Backward-chaining auf alle möglichen Lösungen oder Teilmengen des Ableitungswissens, die allein durch rückwärtige Verkettung über gemeinsame Daten in Prämisse bzw. Schlußfolgerungsteil nicht in den Suchraum gelangen können. Dies ist aber häufig wünschenswert: wird bspw. im Rahmen der

Absatzanalyse in einem diagnostischen Führungsinformationssystem "bekannt", daß sich Währungsparitäten dramatisch verändert haben, könnten so automatisch, d.h. ohne explizite Vorgabe dieses Ziels, Konsequenzen für die konsolidierte Konzernbilanz prognostiziert werden.

Die Verbindung dieser in NexpertObject als *knowledge-islands* bezeichneten Regelbäume kann explizit und lokal geschehen, z.B. durch eine if-changed-Prozedur, oder global durch Kontrollparameter. Abb. 6-18 zeigt solche Kontrollparameter am Beispiel des Strategiemenüs von NexpertObject. Die ersten fünf Parameter (Forward ...) verbinden knowledge-islands, indem automatisch mögliche Konsequenzen für per Rückwärtsverkettung erschlossene Fakten durch Vorwärtsverkettung untersucht werden. Dabei kann nach Schlußfolgerungstyp (Hypothese oder Seiteneffekt/Aktion) und Ergebnis der Schlußfolgerung differenziert werden. Die anderen Kontrollparameter repräsentieren globale Einstellungen für das Schließen durch Vererbung, die über Meta-Slots lokal individuell geändert werden können, wie bereits in Kapitel 6.3.2.1 beschrieben. Darüber hinaus können alle Strategieparameter während des Problemlösungsprozesses dynamisch durch Seiteneffekte in Regeln oder zugeordnete Prozeduren in Frames geändert werden.

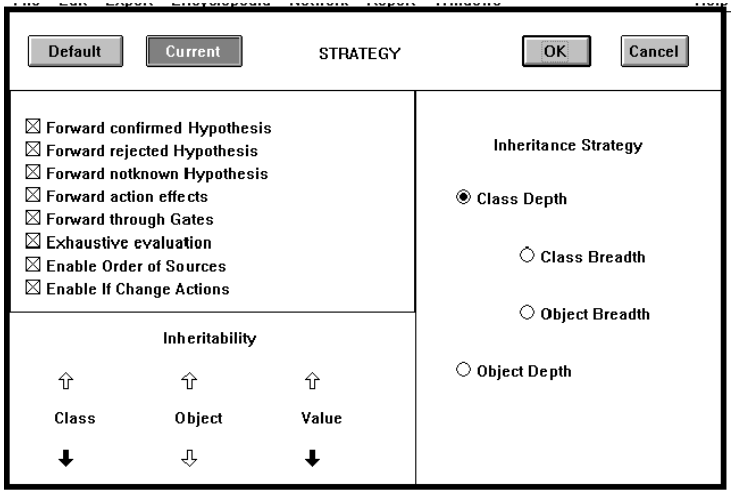


Abb. 6-18: Kontrollstrategien in NexpertObject

Alle bisher beschriebenen Problemlösungsmethoden gehen davon aus, daß Wissenszuwächse dauerhaft und endgültig sind. Dies trifft jedoch in der Praxis selten zu. Neue oder veränderte Fakten erzwingen die (teilweise) Rücknahme getroffener Schlußfolgerungen und das Treffen neuer (korrigierter) Entscheidungen. Theoretische Untersuchungen dieses als nicht-monotones Schließen bezeichneten Komplexes finden sich bspw. in [Brewka 89, Non Monotonic Logics] und [Gnesereth 87, Logical Foundations], Kap. 6. In der Praxis werden Systeme, die in der Lage sind Schlußfolgerungen zurückzunehmen, als Belief-Revision- oder allgemeiner Truth-Maintenance-Systeme (TMS) bezeichnet. Da selten alle bisher getroffenen Schlußfolgerungen von der Änderung betroffen sind, wurden verschiedene Ansätze für eine effiziente Revision entwickelt. Die einfachste Form stellt das *Backtracking* dar, bei dem basierend auf einer Protokollierung der Inferenzen alle Schlüsse neuberechnet werden, die nach der erstmaligen Verwendung des geänderten Datums gezogen wurden. Diese in der Sprache Prolog standardmäßig enthaltene Strategie erfaßt aber im Zuge der Regelverkettung auch Schlußfolgerungen, deren Ergebnis dadurch gar nicht verändert wird. Effizientere Ansätze verwenden anstelle eines einfachen Inferenzprotokolls individuelle Begründungen je Schlußfolgerung. Bei Rücknahme eines Faktums können dann nur noch diejenigen Inferenzen revidiert werden, deren Begründungen ungültig geworden sind. Nach der Art der verwendeten Begründung werden folgende zwei Haupttypen unterschieden:

- *Justification-Based-Truth-Maintenance-Systeme (JTMS)*
  Als Begründungen werden hier die unmittelbaren Vorbedingungen verwendet, d.h. entweder faktische Basisannahmen am Ende von Regelbäumen oder erschlossene Fakten aus vorgelagerten Regeln. Implementierungsbeispiele sind in [Doyle 79, TMS], [McAllister 80, TMS] und [Goodwin 82, TMS] beschrieben.
- *Assumption-Based-Truth-Maintenance-Systeme (ATMS)*
  Hierbei dienen ausschließlich Basisannahmen als Begründung, d.h. evtl. vorgelagerte Regeln werden auf deren Basisannahmen "reduziert". Ein Implementierungsbeispiel findet sich in [deKleer 86, ATMS].

Neben den genannten Beispielen finden sich Konstrukt-mäßige Implementierungen in generelleren Entwicklungswerkzeugen nur selten und mit starken Einschränkungen, so z.B. in KEE und KnowledgeCraft (vgl. [Puppe 91, Expertensysteme], S. 64).

6.3.4. Anwendungs-Spezifikation

6.3.4.1. Allgemeine Anwendungs-Konstrukte

6.3.4.1.1. Konsultation der Wissensbasis

Die Initiierung der Problemlösungskomponente in Bezug auf eine konkrete Fragestellung, d.h. die Konsultation der Wissensbasis, ist eng mit den verwendeten Formalismen der Wissensrepräsentation verbunden. Bei regelbasierten Systemen werden Kommandos zum Starten der Vorwärts- oder Rückwärtsverkettung auf Basis vorgegebener Ausgangswerte oder Zielhypothesen zur Verfügung gestellt. Bei (rein) objektorientierten Systemen ist eine Initialisierungs-Message zur Ausführung einer Startmethode an ein Objekt der Wissensbasis zu richten.

6.3.4.1.2. Rollback-Funktion

Im Gegensatz zu konventionellen DSS-Generatoren hängt der Verlauf der Problemlösung, d.h. das verwendete und zu einer Art "temporärem Programm" zusammengestzte Ableitungswissen, in höherem Maße von Dialogeingaben des Benutzers ab. In Verbindung mit der Erklärungskomponente, die dazu dient, diesen Entscheidungsprozeß jederzeit während der Problemlösung transparent zu machen, ist es notwendig, ausgewählte Eingaben der bisherigen Konsultation zu korrigieren und dann auf dieser Basis die Suche nach einer Problemlösung erneut aufzunehmen bzw. fortzusetzen. Werkzeuge wie ESE oder PersonalConsultant Plus bieten hierbei das Konstrukt der Rollback-Funktion an, mit der alle bis zu einem bestimmten Zeitpunkt getätigten Benutzereingaben bestätigt oder zurückgenommen bzw. genauer "freigestellt" werden können; bei dieser technisch einfachen Lösung werden nicht die von den Korrekturen abhängigen Schlußfolgerungen gezielt im Sinne eines Backtracking zurückgenommen, sondern eine komplett neue Konsultation vom Punkte Null gestartet; vor wiederholter Benutzeranfrage wird lediglich auf bereits früher getätigte und nicht zurückgenommene Benutzereingaben zurückgegriffen. Während im Falle von PersonalConsultant Plus beliebige Eingaben, insbesondere auch nicht unmittelbar aufeinanderfolgende korrigiert werden können, muß bei ESE auf einen Punkt bis dahin zusammenhängend beibehaltener Eingaben zurückgeschritten werden.

Insgesamt geht dabei die zuvor ermittelte Lösung verloren, so daß insbesondere Konstrukte in Richtung auf eine Sensitivitätsanalyse der Wissensbasis derzeit praktisch fehlen bzw. im Einzelfall vom Entwickler implementiert werden müssen.

6.3.4.1.3. Falldatenbank

Die Möglichkeit, einzelne Konsultationsergebnisse analog zu dem Cases-Konstrukt konventioneller DSS-Generatoren zum Zwecke des späteren Fallvergleichs abzuspeichern, ist nur in Verbindung mit Entwicklungswerkzeugen mit fallbasiertem Schließen möglich. Der Fokus liegt dabei jedoch weniger auf dem Fallvergleich, denn auf der Beschleunigung der Problemlösung durch Identifikation einer möglichst ähnlichen Konsultation, die daran anschließend durch wenige Modifikationen und vor allem durch weniger Schlußfolgerungsschritte und damit auch Benutzerdialog-Aktionen zu einer individuellen Lösung modifiziert wird.

6.3.4.1.4. Erklärungskomponente

Die Erklärungskomponente stellt eine charakteristische, allgemeine Anwendungsfunktion dar, die von allen Werkzeugen konstrukt-mäßig unterstützt wird. In ihrer einfachsten und zumeist (auch nur) angebotenen Form besteht sie aus einer aufbereiteten Protokollierung der einzelnen Inferenzschritte (Trace) der Problemlösungskomponente, d.h. z.B. welche Regeln in welcher Reihenfolge mit welchem Ergebnis ausgewählt, getestet und ausgeführt oder verworfen wurden.

Diese (schwache) Form der Erklärung gehört nach Puppe (vgl. [Puppe 91, Expertensysteme], S. 132) zu der Gruppe der *direkten* Erklärungen, die unmittelbar aus dem Programmcode abgeleitet werden. Bezüglich der Fragestellungen lassen sie sich in die beschriebenen WIE-Fragen für den Hergang einer Schlußfolgerung und WARUM-Fragen als Rechtfertigung gestellter Aufforderungen zu Benutzereingaben unterscheiden. Direkte Erklärungen werden dynamisch und kontextabhängig generiert. Bei WIE-Fragen werden das aktuell verfolgte Ziel und der nicht per Inferenz bestimmbare und deshalb im Dialog erfragte Parameter angezeigt.

Davon zu unterscheiden sind *indirekte* Erklärungen, die zumeist als freie, textuelle Rechtfertigungen oder Kommentierungen des Systementwicklers (Knowledge Engineer) oder (Fach-)Experten dienen. Ein Beispiel für diese statischen Erklärungen stellt das *Justification*-Konstrukt von ESE dar.

Das für Erklärungskomponenten relevante Wissen unterscheidet Clancey in (vgl. [Clancey 83, Framework for Explanation]) drei Haupttypen:

- ❒ *strategisches Wissen* zur Rechtfertigung der Reihenfolge von Inferenzen; dies entspricht dem in dieser Arbeit verwendeten Inhalt des Kontrollwissens.

- ❒ *strukturelles* Wissen über die Beziehungen der Wissensbasis-Objekte; dies korrespondiert mit dem Ableitungswissen.
- ❒ *unterstützendes* Wissen für indirekte Erklärungen; dies fällt unter Faktenwissen in der hier verwendeten, funktional-konstruktmäßig orientierten Terminologie.

Wesentlich für die Akzeptanz von Erklärungskomponenten ist die Berücksichtigung des Vorwissens des Anwenders. Dadurch können "selbstverständliche" Tatbestände ausgefiltert werden. Ansatzmöglichkeiten zur Realisierung dieses besonders für Führungsinformationssysteme relevanten Aspekts bieten Benutzermodelle (vgl. [Kobsa 85, Benutzermodellierung]). Indirektes oder unterstützendes Wissen stellt den Bezug zu weiteren Anforderungen von Führungsinformationssystemen her: so können Lexika zur Erklärung der Fachterminologie definitorische Zusammenhänge auf Anfrage erläutern, Referenzen auf fachlich zuständige Experten die Verbindung zu Ansprechpartnern herstellen oder auf relevante Problembereiche verweisen, alles Facetten zur Erhöhung des Informationsgehalts entsprechend den Ausführungen in Kapitel 3.5.3.4.

Weitere vom Problemtyp abhängige Anforderungen und Realisierungsmöglichkeiten der Erklärungskomponente finden sich bspw. in [Puppe 91, Expertensysteme], S. 136-138. Ihrer konstrukt-mäßigen Implementierung zur Verbesserung der Akzeptanz von Expertensystemanwendungen wird hohe Bedeutung beigemessen (vgl. [Kurbel 89, Expertensysteme], S. 29).

6.3.4.2. Problemspezifische Anwendungs-Konstrukte

6.3.4.2.1. Diagnostik

Hauptanwendungsgebiet wissensbasierter Systeme ist die Diagnostik bzw. Klassifikation. Puppe charakterisiert diesen Problemtyp wiefolgt (vgl. [Puppe 91, Expertensysteme], S. 74):

- ❒ Zwei disjunkte Mengen von Merkmalen (Symptome) und Lösungen (Diagnosen), verbunden durch unsicheres Beziehungswissen beschreiben den Problembereich.
- ❒ Gesucht werden eine oder mehrere Diagnosen für eine (unvollständig) gegebene Teilmenge von Symptomen, wobei auch zusätzliche, zu Beginn nicht gegebene Symptome auf ihre Relevanz zu untersuchen und nachzufordern sind.

Während die Ursprünge diagnostischer Expertensysteme im medizinischen Bereich liegen, wurden die größten praktischen Erfolge bei technischen Systemen erzielt, z.B. im Rahmen der Fehlersuche bei Qualitätskontrolle oder Reparaturen, der Überwachung von Fertigungsprozessen sowie Selektionsproblemen. Dies liegt zu einem großen Teil daran, daß Sympto-

me (Meßdaten) und Symptom-Diagnose-Beziehungen im technischen Bereich eher deterministisch und sicher sind.

Diese Problemcharakteristika finden sich qua definitione auch in Führungsinformationssystemen: Symptome entsprechen beobachteten unternehmensinternen und -externen Kenngrößen, Diagnosen der daraus abgeleiteten, (standardisierten) Situationsbewertungen und Handlungsempfehlungen. Beispiele sind etwa Produktklassifikationen in einer Marktattraktivitäts-Marktanteils-Portfolio-Matrix, Liquiditäts- und Bilanzanalysen oder Einstufungen der Ertrags- und Innovationskraft eines Unternehmens. Das Nachfordern zusätzlicher Symptomwerte entspricht der in Kapitel 5.3.4 gestellten Anforderung nach "prospektiver Disaggregation". Demzufolge sind auch die speziell für den Problemtyp Diagnostik/Klassifikation kreierten Konstrukte bzw. Konstrukt-Kombinationen in spezialisierten, wissensbasierten Entwicklungswerkzeugen von besonderer Relevanz für wissensbasierte Komponenten in Führungsinformationssystemen.

Bei sicheren Merkmalen und eindeutigen Merkmals-Lösungs-Beziehungen können Entscheidungsbäume und -tabellen mit Vorwärtsverkettung eingesetzt werden (sichere Klassifikation). Für definitorische Zusammenhänge, die z.B. häufig als Kennzahlensysteme eine Basiskomponente von Führungsinformationssystemen bilden, eignen sich modellbasierte Ansätze: bei der *überdeckenden Klassifikation* wird die Sichtweise zwischen Merkmalen und Lösungen aufgrund einer unterstellten "verursachenden" Wirkung einer Lösung auf Merkmale quasi "umgedreht", d.h. es wird nach Lösungen gesucht, die die beobachteten Merkmale am besten "erklären"; die *funktionale Klassifikation* basiert auf Modellen mit Komponenten zur Transformation von Materialien, deren Fehlverhalten sich an anormalen Umwandlungszuständen der Materialien festmachen läßt. Repetitive Problemfelder mit wenigen, ähnlichen Lösungen legen Falldatenbanken und Zusatzwissen für Ähnlichkeitsvergleiche nahe (vgl. [Puppe 90, Problemlösungsmethoden], S. 114 ff.).

Eine direkte (einstufige) Zuordnung von Symptomen zu Diagnosen ist aus inhaltlichen und Effizienzgründen jedoch in den seltensten Fällen praktikabel. Vielmehr empfiehlt sich ein Lösungsprozeß in Zwischenstufen, insbesondere bei unvollständigen und unsicheren (Ausgangs-)merkmalen und Beziehungen. Hierfür wird von Puppe das Modell des *diagnostischen Mittelbaus* (vgl. [Puppe 90, Problemlösungsmethoden], S. 58) eingeführt, das auf allgemeinere Arbeiten von Clancey zur *heuristischen Klassifikation* zurückgeht (vgl. [Clancey 85, Heuristic Classification]). Dabei wird der Problemlösungsprozeß in zwei Phasen aufgeteilt (vgl. Abb. 6-19):

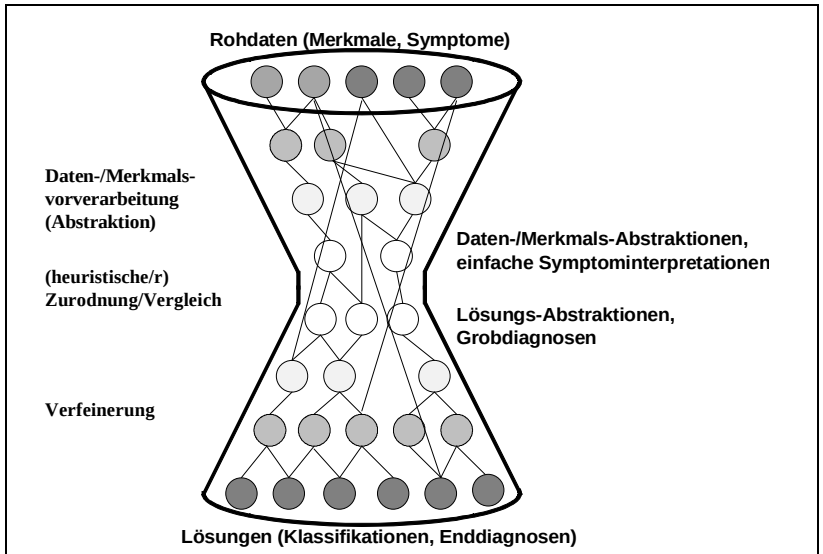


Abb. 6-19: Problemlösungsphasen des diagnostischen Mittelbaus

- ❒ eine (komprimierende) Daten- bzw. Merkmalsvorverarbeitung, bei der Rohdaten (Symptome) in mehreren Stufen in abstraktere Situationsbeschreibungen überführt werden, die meist der Fachsprache des Experten entlehnt sind, und
- ❒ eine (expandierende) diagnostische Auswertung, bei der die abstrakten Symptominterpretationen durch heuristisches Zuordnungswissen mit Grobdiagnosen bzw. Lösungsklassen in Verbindung gebracht werden, die dann im Rahmen einer vorgegebenen baum- oder netzartigen Lösungsstruktur zu einer oder mehreren Diagnosen (Lösungen) verfeinert werden.

Neben den elementaren Mechanismen der Vorwärts- und Rückwärtsverkettung wurden zur Lösungssuche bei unvollständiger bzw. unsicherer Wissensbasis verfeinerte, übergeordnete Such-Strategien entwickelt. Bei der *Establish-Refine*-Strategie wird eine strenge Lösungshierarchie, bei der eine Lösung(sklasse) jeweils nur eine (abstraktere) Lösungsklasse als Vorgänger hat, schrittweise abgearbeitet, indem auf jeder Hierarchiestufe zunächst eine Lösungsklasse evtl. durch Rückwärtsverkettung (Nachfragen) bestätigt (etabliert) wird, bevor von ihr ausgehend zur nächstniedrigeren Hierarchiestufe gewechselt wird. Bei der

Hypothesize-and-Test-Strategie werden dagegen durch Vorwärtsverkettung zunächst in Frage kommende Lösungen (am Ende des Lösungsraums) als Verdachtshypothesen generiert, die dann gezielt, d.h. bezüglich ihrer Erfolgswahrscheinlichkeit bzw. ihrer Überprüfungskosten geeignet bewertet, durch Rückwärtsverkettung (Nachfragen) überprüft werden.

6.3.4.2.2. Konstruktion

Diagnostik bzw. Klassifikation setzen eine bekannte Menge möglicher Lösungen voraus, aus der (lediglich) die geeignetsten ausgewählt werden. In Führungsinformationssystemen kann dies zwar für Situationsbewertungen im Rahmen standardisierter Fälle brauchbar eingesetzt werden, darüber hinaus sind jedoch oft auch individuelle (neue) Lösungen zu generieren, z.B. in Form eines aktuellen, dynamischen Ausnahmeberichts oder eines flexibel aus parametrisierten Basisberichten konfigurierten Statusberichts oder Berichtssystems, einer kontextabhängigen Tagesordnung oder problemspezifischen Liste einzuladender Fachspezialisten bis hin zur Generierung von Handlungsplänen. Die Problemlösung ist dabei aus Lösungselementen zusammenzusetzen. Dies entspricht dem Problemlösungstyp der Konstruktion mit folgenden drei Untertypen:

- *Konfigurierung*
  Hierbei werden vorgegebene, atomare Bausteine selektiert, parametrisiert und in mehreren Stufen aggregiert. Das Resultat muß geforderte Eigenschaften sowohl der Atome als auch des Aggregats und beliebiger Zwischenaggregate (Moleküle) erfüllen. Als Anwendungsbeispiel bei Führungsinformationssystemen läßt sich die Konfigurierung eines Monatsberichts vorstellen: atomare Bausteine können auf verschiedensten Aggregationsniveaus gebildet werden, z.B. niedrig als Kennzahlen mit Ist-, Plan- sowie prozentualer und absoluter Abweichung oder hoch als Umsatz-, Liquiditäts- oder Kapazitätsauslastungsberichte. Eigenschaften sind sowohl an den Basiselementen fest zu machen, z.B. als (numerische) Grenzwerte einzelner Kennzahlen, oder an den Aggregaten, z.B. als "die zehn umsatzstärksten Produkte".
- *Zuordnung*
  Im Gegensatz zur Konfigurierung operiert die Zuordnung mit Objekten, deren Eigenschaften bereits vollständig spezifiziert sind. Die Aufgabe besteht darin, Objekte einer Menge in eine andere abzubilden. Handelt es sich bei der Zielmenge um Zeitintervalle, wird vom Spezialfall des *Scheduling* gesprochen. Die einzuhaltenden Eigenschaften (Rahmenbedingungen) beziehen sich ausschließlich auf diese Abbildungsfunktion bzw. deren Resultat. Häufig angeführtes Beispiel ist die Erstellung von Stundenplänen. Für den Bereich der Führungsinformationssysteme lassen sich kaum sinnvolle Anwendungen finden.

- *Planung*
  Hierbei treten zeitliche Reihenfolge und Operatoren, die die Attribute der Basiselemente verändern, hinzu. Klassisches Beispiel ist die Arbeitsplanung von Werkstücken. Bei Führungsinformationssystemen lassen sich potentielle Anwendungen vor allem in vor- bzw. nachgelagerten Bereichen vorstellen, z.B. als Unterstützung einer Maßnahmenplanung oder dem Entwurf von Berichtssystemen, bestehend aus verschiedensten Berichten und Adressaten zu abgestimmten Zeitpunkten. Im letzten Fall repräsentieren die Operatoren Aufgaben und Funktionen der Berichtsgenerierung.

Zur Bewältigung des typischerweise großen, exponentiell wachsenden Lösungsraumes von Konstruktionsproblemen (vgl. [Nilsson 82, Artificial Intelligence], Kap. 2 und 3) wurden spezielle Kontrollstrategien entwickelt, die sich wiefolgt aus Basismethoden zusammensetzen:

- Beim *Skelett-Konstruieren* wirken Abstraktion, Modularisierung und Hill-Climbing-Technik zusammen. Abstraktion und Modularisierung bedeuten in Analogie zur Establish-Refine-Strategie, daß zunächst "aggregierte" Elemente, z.B. Teilberichte, in die Breite ausgewählt und zusammengesetzt werden, bevor diese (Teilkonstruktionen) dann verfeinernd in die Tiefe "(aus)konfiguriert" werden. Der Lösungsraum, d.h. das Konstruktionswissen ist dabei als hierarchischer Und-Oder-Graph vorzugeben. Welche der Oder-Verzweigungen, d.h. der Konstruktionsalternativen, greifen, und wie damit die Expansion des Graphen aussieht, wird durch heuristische Regeln (Hill-Climbing) gesteuert. Damit entspricht das Skelett-Konstruieren einer multiplen heuristischen Klassifikation (vgl. [Puppe 90, Problemlösungsmethoden], S. 139). In der allgemeineren Form des *Generate-and-Test* entfällt die explizite Benennung der Objekte(benen) (vgl. [Puppe 90, Problemlösungsmethoden], S. 142).
- Die *Vorschlagen-und-Verbessern*-Strategie verbindet Vorwärtsverkettung, Überwachung mit Constraints und abhängigkeitsgesteuertes, wissensbasiertes Rücksetzen (vgl. [Puppe 90, Problemlösungsmethoden], S. 146). Sie dient primär komplexen Parameter-Einstellungen von Konfigurierungsproblemen; das Äquivalent für Zuordnungsprobleme ist die *Vorschlagen-und-Vertauschen*-Strategie, bei der Inkonsistenzen bzw. fehlende Optimalität einer durch Vorwärtsverkettung gewonnenen Ausgangslösung schrittweise durch das Auswechseln von Lösungselementen anstatt durch das Verändern von Parameterwerten derselben erreicht wird.
- Die *Least-Commitment*-Strategie schließlich basiert auf vereinfachenden Umformungen des Problems und Constraint-Propagierung. Bei der Lösungssuche wird das Prinzip verfolgt, geringst mögliche Festlegungen von Elementen und Parametern zu treffen. Bei

einer Berichtskonfigurierung würden also bspw. nicht sofort die aufzunehmenden Kennzahlen aufgrund von Grenzwertüberschreitungen (namentlich) festgeschrieben.

Konstrukt-mäßige Implementierungen der beschriebenen Strategien finden sich hauptsächlich in Spezialanwendungen. Zunehmend wird jedoch versucht, die Erkenntnisse auf Problemtyp-spezifische Entwicklungswerkzeuge zu verallgemeinern. Ein Beispiel hierfür ist PLAKON ([Neumann 87, Wissensbasierte Planung]; [Cunis 89, Construction]).

6.3.4.2.3. Simulation

Simulation beschränkt sich in wissensbasierten Systemen bisher auf relativ einfache, eng begrenzte Anwendungsbereiche, z.B. die Überprüfung diagnostizierter oder konstruktiver Problemlösungen. Auch die in Entwicklungswerkzeugen implementierten Konstrukte für Wissensrepräsentation und Problemlösung stellen nur eine kleine Auswahl der aus dem Operations Research bekannten Verfahren dar. Die Ursache kann darin gesehen werden, daß Simulation umfangreiche Modelle über die Beziehungszusammenhänge erfordert. Bei Führungsinformationssystemen ist dieses Wissen typischerweise in konventionellen Analyse- und Planungsmodellen abgelegt, also außerhalb möglicher ergänzender wissensbasierter Komponenten. Simulation dürfte sich somit primär als Alternativ- bzw. Prognose-Rechnung extern abspielen, bevor eine (diagnostizierende) Bewertung durch ein aufgesetztes Expertensystem erfolgt. Aus diesen Gründen wird hier auf eine tiefergehende Betrachtung von Konstrukten wissensbasierter Simulation verzichtet. Ausführliche Darstellungen finden sich bspw. in [Puppe 91, Expertensysteme], S. 102-111, [Puppe 90, Problemlösungsmethoden], S. 182-198 und in [Round 89, KB-Simulation].

7. Leistungsdefizite und Entwicklungsbedarf

7.1. Evaluations-Szenario

7.1.1. Kriterien

Die in Kapitel 6 durchgeführte Konstruktanalyse von Werkzeugen zur rechnergestützten Gestaltung von (Management-) Arbeitsplätzen bestätigt je Werkzeugklasse ein bereits relativ hohes Leistungsniveau. Somit kann angenommen werden, daß eine Integration von Anwendungen verschiedener Werkzeugklassen sowohl bezüglich Produktivität für den Entwickler als auch bezüglich Leistungsspektrum für den Anwender bessere Ergebnisse liefert als eine Ergänzung einer Werkzeugklasse um Funktionalitäten der anderen Klassen. Als Beleg dafür können bspw. die Worksheet- und neuerdings sogar What-if-Komponenten des EIS-Generators Pilot genannt werden, deren Funktionsspektrum und Bedienerfreundlichkeit weit hinter denen von Planungssprachen oder Tabellenkalkulationssystemen zurückbleiben. Umgekehrt hinken die (grafischen) Präsentations-, Selektions- und Verknüpfungsmöglichkeiten von Planungssprachen weit hinter denen von EIS-Generatoren her. Trotz erheblicher Verbesserungen insbesondere bei Tabellenkalkulationssystemen und wissensbasierten DSS-Generatoren unter MS-Windows sind nach wie vor so wesentliche Funktionen wie Integration verschiedener Informationsquellen und Ausnahmeberichtswesen nur mit hohem Eigenentwicklungs- und Wartungsaufwand zu realisieren. In jedem Falle müssen Informationen anderer Anwendungssysteme zunächst in die jeweiligen Datenstrukturen transformiert werden.

Die folgende Analyse von Leistungsdefiziten orientiert sich deshalb neben nach wie vor bestehenden Mängeln der beschriebenen innovativen Repräsentanten je Werkzeugklasse an deren Kooperationsfähigkeit im Sinne des in Kapitel 4 dargestellten, verteilten Konzepts rechnergestützter Arbeitsplätze. Zur Illustration werden die Kritikpunkte anhand einer umfangreichen, damit praxisrelevanten Fallstudie exemplifiziert (vgl. Kapitel 7.1.2). Als Bewertungskriterien wird auf ausgewählte, in Kapitel 3.5 dargestellte kritische Erfolgsfaktoren technischer Art zurückgegriffen:

- ❒ Struktur und Funktionalität
- ❒ Verteilung und Integration
- ❒ Entwicklungs- und Wartungs-Produktivität

7.1.2. Fallstudie

Die im folgenden beschriebene Fallstudie basiert auf den prototypischen Entwicklungsarbeiten eines generischen Analyse- und Planungsmodells für den Bereich des Versicherungs-Controlling (vgl. [Reichardt 92, Versicherungs-Controlling]). Das ursprünglich dort mit MS-Excel implementierte Modell wurde zum Zwecke der hier angestrebten Anlayseziele in die in Abb. 7-1 dargestellte Architektur eines rechnergestützten Arbeitsplatz-Systems erweitert und integriert.

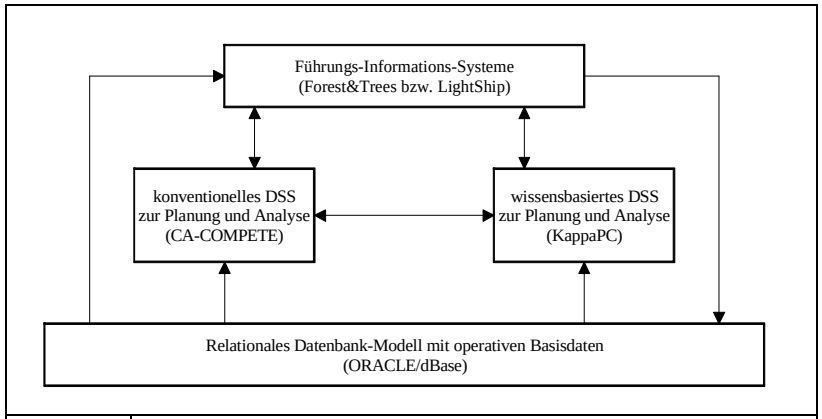


Abb. 7-1: Komponenten der verteilten RAP-Fallstudie

Dieses besteht aus den folgenden Komponenten:

- Einem relationalen Datenbankmodell zur Versorgung aller anderen Komponenten mit Basisdaten der operativen Unternehmensebene. Als Werkzeug werden ORACLE bzw. dBASE eingesetzt.
- Ein konventionelles Decision-Support-System (DSS) zur Planung und Analyse von Vertrieb, Prämienkalkulation und Erfolgsrechnung. Das DSS selbst besteht aus 3 Teilmodellen, die untereinander dynamisch verknüpft sind. Eine detaillierte Beschreibung erfolgt im Kapitel 7.3. Als Entwicklungswerkzeug wird die Planungssprache CA-Compete eingesetzt.

- Einem wissensbasierten DSS (Expertensystem, XPS) zur Unterstützung analytischer (Bewertung der Unternehmenssituation) und planerischer (Prämienkalkulation) Aufgaben. Als Entwicklungswerkzeug wird der XPS-Generator KappaPC eingesetzt.
- Einem Führungsinformationssystem (Executive Information System, EIS), das Führungskräften den Zugang zu Basis- und abgeleiteten Informationen der beiden DSS bzw. weiteren, relevanten Informationsquellen ermöglicht sowie es gestattet, über Dialogfunktionen unternehmerische Entscheidungsparameter zu modifizieren und die analytischen und planerischen Unterstützungsfunktionen der DSS über eine gemeinsame, leicht bedienbare Benutzeroberfläche zu nutzen. Als Entwicklungswerkzeuge kommen die EIS-Generatoren Forest&Trees und LightShip zum Einsatz.

7.2. EIS-Generatoren

7.2.1. Struktur und Funktionalität

Für eine hohe Anwenderakzeptanz steht eine weitgehende Kongruenz zwischen präsentierter, abrufbarer bzw. gestaltbarer Informationskomplexität und mentalem Modell im Vordergrund. D.h., Informationen müssen als komplexe Konglomerate verschiedenster Informationselemente individuell gebildet, verknüpft und erweitert werden können. Die assoziative Verknüpfung darf nicht auf bestimmte, z.B. numerische Datentypen, identische Informationsarten, z.B. Umsatzzahlen verschiedener Aggregationsstufen oder deren rechnerische Beziehung, z.B. Absatzmenge und -preis mit Umsätzen beschränkt sein. Vielmehr müssen unterschiedlichste Informationsarten mit deren Bearbeitungsfunktionen, z.B. Umsätze mit Fertigungsdaten, Produktinformationen, Markt- und Vertriebsdaten usw. integriert werden.

Ferner darf die Verknüpfung nicht statischen Charakters sein, sondern muß sich dynamisch und leicht dem Analyse- bzw. Untersuchungsziel des Anwenders anpassen lassen. Dieses Betrachtungsziel wiederum hängt entscheidend von den Ursachen des behandelten Problems ab. Somit sind die im Einzelfall benötigten bzw. gewünschten Verknüpfungen im Sinne eines Drill-Down weder a priori noch für längere Zeit bestimmbar bzw. festzulegen.

Als Beispiel seien Umsatzabweichungen zwischen Ist und Plan auf aggregiertem Niveau angeführt. Mögliche Ursachen könnten sein:

❒ Engpässe bei maschinellen oder personellen Fertigungskapazitäten

❒ Engpässe bei der Versorgung mit Zulieferteilen

❒ Engpässe in der Distributionslogistik

- ❒ Aktivitäten der Konkurrenz, z.B. punktuelle oder globale Werbemaßnahmen, Preissenkungen oder Produktneueinführungen
- ❒ Fehleinschätzungen in der Planung, z.B. saisonaler Aspekte usw..

Im ersten Schritt einer Lokalisation des Problems (*Problemidentifikation*) kann die von EIS-Generatoren angebotene Drill-Down-Funktionalität über Informationen gleichen Typs eingesetzt werden, im vorliegenden Falle also der Umsatzzahlen verschiedener Aggregationsniveaus. Auf diese Weise lassen sich etwa zeitliche, regionale oder produktmäßige Eingrenzungen vornehmen. Dieser Suchprozeß kann durch ein selektives Exception-Reporting auf der Basis vorgegebener Toleranzwerte beschleunigt werden. Voraussetzung für eine Praxistauglichkeit ist jedoch, daß die Funktionen Selektion und Weiterverarbeitung sowie die Individualität der Grenzwerte *zusammen* verfügbar sind. Genau dies ist bei den angebotenen Werkzeugen jedoch (noch) nicht gegeben:

- ❒ Zwar wird in einem Fall die Selektion von Abweichungen aller Aggregationsstufen in Form eines dynamischen Auswahlmenüs für weitere Verzweigungen geboten, es kann aber nur *ein globales* Grenzwertpaar je Informationsart, hier Umsatz, definiert werden (Beispiel Pilot). Damit kann eine differenzierte Bewertung bspw. von geringwertigen oder anteilsmäßig unbedeutenden Produkten oder Regionen nicht abgebildet werden.
- ❒ Im anderen Extrem sind zwar differenzierte Grenzwerte je Aggregationsstufe, z.B. Produkt- oder Regionsgruppe möglich, dafür fehlt die automatische Selektion gänzlich (Beispiel CommanderEIS), oder sie ist auf eine reine Anzeige von Hinweistexten ohne direkte Weiterverarbeitungsmöglichkeit beschränkt (Beispiel Forest&Trees).

Im zweiten Schritt der *Ursachenanalyse* treten die Schwächen heutiger Drill-Down-Konstrukte zu Tage. Zunächst müssen Verzweigungen zu in der Regel anderen Informationsarten und -quellen hergestellt werden können, die potentiell als Auslöser, Verursacher oder erklärender Faktor in Frage kommen. Im Beispiel sind dies Fertigungspläne, Lagerbestände, Vertriebs- oder Marktdaten. Für diese wiederholt sich gegebenenfalls die beschriebene (homogene) Drill-Down- und Exception-Reporting-Funktionalität iterativ. Die individuelle Selektion relevanter, d.h. als Erklärung in Frage kommender Elemente, umfaßt in der Regel die Ergänzung mit textueller, kommentierender Information. Die so gewonnenen, in sich wieder verknüpften Informationseinheiten müssen schließlich im Sinne einer aktuellen, dynamischen Berichterstattung in ein vom Anwender abrufbares (heterogenes) Drill-Down-Netzwerk überführt werden können.

Im Ergebnis könnte das zu generierende Berichtssystem dem Anwender dann im angeführten Beispiel zunächst eine Übersicht kritischer Abweichungen liefern, z.B. Planabweichungen

des Umsatz in der Region Ost für alle Produkte und des Umsatz für eine fertigungstechnisch zusammengehörige Produktgruppe für alle Regionen. Im ersten Falle könnte die Drill-Down-Funktionalität neben einem erläuternden Kommentar des regionalen Vertriebsleiters je nach analysierter Ursache Vergleichsstatistiken der Absatzzahlen, Vertreterdichte oder Werbebudgets mit der Konkurrenz abrufbar bereithalten. Im zweiten Falle könnten Kapazitätsdiagramme oder Produktinformationen mit Stücklisten und Lagerumschlagsstatistiken oder Lieferzeitanalysen kritischer Zulieferteile angebunden sein.

Die Realisierung dieser Funktionalitäten erfordert Fähigkeiten der Entwicklungswerkzeuge in zweierlei Hinsicht:

- ❒ Zum einen müssen Informationselemente verschiedenster Art, hier Texte und Tabellen bzw. Grafiken unterschiedlicher Datenquellen, zu gemeinsam selektierbaren, komplexen Einheiten zusammengefaßt werden können, ohne daß sie die Drill-Down-Funktionalität ihrer Elemente verlieren.
- ❒ Zum anderen müssen diese in sich komplexen Einheiten untereinander beliebig verknüpfbar sein, ohne daß das Springen zwischen den Konglomeraten mit einem "Überschreiben" der bisherigen Ansicht verbunden ist. D.h., die Konglomerate oder Informationsobjekte müssen sich als individuell positionier- und manipulierbare "Fenster" auf dem Bildschirm präsentieren.

Auch diese Eigenschaften sind trotz punktueller Fortschritte in ihrer Gesamtheit nicht als generische Konstrukte in heute verfügbaren Entwicklungswerkzeugen (EIS-Generatoren) zu finden. Die geforderten Effekte lassen sich in einzelnen Ausnahmefällen zwar durch ergänzende Programmierung implementieren (Beispiel Pilot), extremer zeitlicher und personeller Aufwand sowie fortbestehende Inflexibilität disqualifizieren diese Lösung jedoch für die Praxis. Die Ursache dafür liegt darin begründet, daß keiner der EIS-Generatoren Information, losgelöst von einer aktuellen Präsentation, konsequent als zusammengesetztes Objekt begreift und abbildet:

- ❒ So wird die Zusammenführung verschiedener Informationsarten und -quellen, z.B. Umsatz- und Fertigungsdaten sowie Kommentaren zu gemeinsam selektierbaren Objekten erst auf der Präsentationsebene vorgenommen, indem entsprechende "Fenster"-Ausschnitte zu strukturell statischen Bildschirmeinheiten spezifiziert werden (Beispiele LightShip, CommanderEIS). Die Verknüpfungen existieren nicht zwischen den Informationselementen selbst, sondern nur auf der Ebene der Bildschirmspezifikation. Dadurch wird ein datengetriebenes Drill-Down unmöglich. Vielmehr muß Drill-Down zwischen Bildschirmen (statisch) spezifiziert werden.

- Zwar erlauben einige EIS-Generatoren die Verlagerung der Informationsverknüpfung auf die Ebene der Datenbasis (Beispiel Pilot), diese Fälle sind jedoch darauf beschränkt, daß die Struktur der verknüpften Informationen identisch ist, d.h. per Drill-Down abgerufene Detailinformationen sich mit derselben Bildschirmstruktur darstellen lassen. Ein (datengetriebener) alternativer Wechsel von Umsatzkennzahlen zu mehreren anders strukturierten Bildschirmen, etwa Fertigungsplänen, Marktstatistiken oder Produktinformationen, läßt sich nicht oder nur mit (inakzeptabel) hohem Programmieraufwand realisieren.
- In den meisten Fällen sind die EIS-Generatoren "Bildschirm"-orientiert, d.h. es kann jeweils nur ein "Informationsobjekt" gleichzeitig betrachtet und untersucht werden. Aufgrund der gleichzeitig relativ starren Struktur dieser Bildschirme ergibt sich ein krasser Widerspruch zu dem essentiellen Erfordernis der Praxis, verschiedene zusammengehörige Informationsobjekte aufgaben-individuell zu "öffnen", zusammenzustellen und zu analysieren. Ansätze zur Beseitigung dieser Schwachstelle beschränken sich auf Popup-Windows für Dialogboxen, Auswahlmenüs oder die Anzeige von Kommentartexten (Beispiele LightShip und CommanderEIS). In beiden Fällen ist jedoch nur jeweils eine zusätzliche Popup-Ebene möglich. Ferner ist dies in "modaler", d.h. die Dialogfunktionen des ersten Bildschirms ausschließender Form, realisiert.
- Dem beschriebenen Anforderungsprofil am nächsten kommende EIS-Generatoren (Beispiel Forest&Trees) arbeiten demgegenüber mit einem View-Konzept. Dabei präsentieren sich zwar alle Informationseinheiten als eigenständige View-Objekte, die beliebig geöffnet, positioniert und manipuliert werden können, ihre Objektstruktur ist jedoch anbieterseitig fest vorgegeben und kann mangels fehlender Spezifikations- oder Programmierschnittstelle auch nicht vom Anwendungsentwickler oder Anwender verändert werden. Immerhin umfassen die View-Objekte als konfigurierbare Elemente ein aus der individuellen Verknüpfungsspezifikation abgeleitetes Drill-Down bzw. Drill-Up, ein individuelles, differenzierbares und nicht auf numerische Aspekte beschränktes Exception-Reporting, das Abweichungen automatisch über die Verknüpfungshierarchie nach oben visualisiert sowie anfügbare textuelle Kommentare. Ferner können über eine integrierte Funktion der View-Objekte mehrere Grafiken oder formatierte Berichte angebunden werden, die sich auch auf andere Informationseinheiten beziehen können.

7.2.2. Verteilung und Integration

Bereits die für Management-Arbeitsplätze typische Eigenschaft der Zusammenführung verschiedenster Informationsquellen impliziert bzw. induziert verteilte, integrative Anwendungskonzepte und -architekturen. Bezüglich (historischer) Ursache und Motivation für das

Fortbestehen und den Ausbau dieser Verteilung können zwei Dimensionen unterschieden werden:

- *organisatorische* Aspekte, wobei aufgrund des Prinzips der Arbeitsteilung (Teil-)daten zeitlich, räumlich und personell getrennt erfaßt, verwaltet und aufbereitet werden. Dies schließt sowohl alle Prinzipien der Organisationsgliederung, z.B. nach Funktionen, Sparten usw. ein, als auch die Arbeitsteilung zwischen Mensch und Maschine (hier: EDV) sowie zwischen Maschinen selbst.
- *(software-)technische* Aspekte, wobei im Hinblick auf Entwicklungs- und Wartungs-Produktivität aufgrund von Datentyp, -struktur und (primärem) Verwendungsziel differenzierte Datenmodelle und Verarbeitungsprogramme zum Einsatz kommen. Hierzu sollen nicht nur aus Datenstruktur oder Verarbeitungsfunktionalität zwingend abzuleitende Differenzierungen gezählt werden, sondern insbesondere auch Argumente der Effizienz und Benutzeradäquatheit eingesetzter Software. Auf diese Weise kommen auch Tabellenkalkulationssysteme als Informationslieferanten in Betracht.

Entwicklungswerkzeuge müssen unter diesen Rahmenbedingungen intern über Integrations- und extern über Kommunikations-Konstrukte verfügen. Bei der Integration steht die Einheitlichkeit der Informationspräsentation und Bedienung, bei der Kommunikation die Minimierung von Daten- und Verarbeitungsredundanzen im Vordergrund. Eine Analyse derzeit angebotener Werkzeuge unter diesen Blickwinkeln offenbart folgende Entwicklungsniveaus und Schwachstellen:

- Im einfachsten Fall wird die Integration bereits auf datentechnischer Ebene realisiert, durch Transformation aller (heterogenen) Informationsquellen in eine (homogene) Datenbasis. Beispiele hierfür sind der EIS-Generator Pilot, die Planungssprachen Compete und und OneUp, aber auch Tabellenkalkulationssysteme wie MS-Excel. Die damit in der Regel verbundene (redundante) Duplizierung des Datenbestandes kann aus Gründen der Effizienz späterer Abfragen (Antwortzeiten) oder damit einhergehender Aggregationen zwar sinnvoll sein, diese Vorgehensweise schränkt jedoch das Spektrum integrierbarer Informationen durch die verfügbaren Datentypen des Zielsystems erheblich ein: so sind Planungssprachen-basierte Systeme auf numerische Werte (z.B. OneUp), Tabellenkalkulations- und Datenbanksysteme auf "flache", zwei-dimensionale Strukturen (z.B. Excel, Pilot) beschränkt. Insbesondere die Integration von Texten, mehrdimensionalen Modellstrukturen und Grafiken bereitet Probleme.

  Verbesserungen können Typ-Erweiterungen bringen, wie bspw. Zeitreihen bei Pilot. Da es sich dabei jedoch um eine Eigenkreation handelt, sind erneut Konvertierungsroutinen zu Fremdformaten, z.B. den Zeitreihenformaten anderer Datenbanksysteme oder Pla-

nungsprachen notwendig. Anders sieht es bei den Dokumenttypen bspw. von CommanderEIS aus, die speziell auf Fremdformate zugeschnitten sind und die funktionale Umsetzung in eine einheitliche Präsentationsoberfläche in sich vereinigen.

Die Verteilungs-Problematik wird in all diesen Fällen in zwei getrennte, zeitlich aufeinander folgende Abschnitte zerlegt, von denen lediglich der erste potentiell Kommunikation mit Fremdsystemen vollziehen kann. Zumeist ist jedoch auch diese Kommunikation auf das Lesen und Kopieren bzw. Konvertieren der Fremddaten beschränkt. Die Kommunikationsprozesse des zweiten Abschnitts mit der dann evtl. ebenfalls verteilt realisierten, intregrativen Datenbasis stellt ein vergleichsweise einfaches, hier untergeordnetes Problem dar, das von den meisten Anbietern gelöst ist.

- Einen qualitativen Fortschritt bringen erst Systeme, bei denen das Kopieren und evtl. Konvertieren von (Fremd-)Daten durch die Spezifikation der Datenbeschaffung ersetzt wird. Beispiele hierfür finden sich bereits in Planungssprachen und EIS-Generatoren: so erlaubt die Planungssprache IFPS, Eingabedaten in Modellen als Abfrage-Link zu internen oder externen Datenbanksystemen zu vereinbaren; beim EIS-Generator CommanderEIS können Berichts- und Grafik-Dokumente in der Briefing-Book-Komponente als Links zu externen Dateien im Netzwerk spezifiziert werden, numerische Abfragen auf OneUp-Planungsmodelle in der ExecuView-Komponente operieren grundsätzlich auf den externen Modelldatendateien. Der EIS-Generator Forest&Trees arbeitet prinzipiell mit Links. Zur Abfragespezifikation wird eine einheitliche, SQL-ähnliche Abfragesprache verwendet, die automatisch verschiedene externe Datenformate umsetzt und in begrenztem Umfang auch Rechenoperationen zuläßt.

  In all diesen Fällen ist die Kommunikation jedoch im wesentlichen auf das "Anzapfen" der externen Datenstrukturen beschränkt. Die externen Anwendungssysteme werden in keinem Fall selbst "aktiv" in dem Sinne, daß sie Anfragen selbst bearbeiten und deren Ergebnisse zurückliefern. Damit steht das "verteilte" Informationsverarbeitungs-Knowhow, wie z.B. Sensitivitätsanalysen oder Zielrechnungen, nicht unmittelbar zur Verfügung. Ein Zugang hierzu ist erst mit zeitlicher Verzögerung nach ablauf-organisatorischen Maßnahmen möglich, die in der Regel auch Wartungsarbeiten am Anwendungssystem zur präsentativen Aufbereitung der zusätzlich generierten Fremddaten einschließen. Die teilweise gewählte Alternative der Reimplementierung dieser weiterverarbeitenden Funktionalitäten im integrierenden Anwendungssystem erscheint aufgrund der bereits eingangs erläuterten Gründe wenig effizient.

- Das maximale Niveau an Verteilung läßt sich erst erreichen, wenn die Kommunikation sich auf fremde Anwendungssysteme ausdehnt. Dies setzt allerdings die Lauffähigkeit der beteiligten Softwaresysteme in einer Multitasking-fähigen Betriebssystemumgebung sowie die bilaterale Existenz gemeinsamer Kommunikations-Protokolle voraus. Beispiele hierfür finden sich mit DDE unter MS-Windows in den EIS-Generatoren LightShip und Forest&Trees, Planungssprachen wie Compete, Tabellenkalkulationssystemen wie MS-Excel und Expertensystemwerkzeugen wie KappaPC.

  Trotz vereinzelter Ausnahmen zur Übertragung von Graphiken, z.B. bei LightShip, lassen sich komplexe, zusammengesetzte Informationsobjekte aufgrund der rudimentären Protokollstruktur nur in mehreren Teilschritten unter großem Entwicklungsaufwand, mit einem hohen Maß an Detailwissen über die Fremdanwendung und teilweise geringer Flexibilität übertragen. Insbesondere sind die DDE-Konstrukte der genannten EIS-Generatoren unmittelbar mit Präsentations-Konstrukten, z.B. Textdokument- oder Grafik-Fenstern, verbunden, aus denen sie zum Zwecke einer datengetriebenen Weiterverarbeitung, z.B. als Auswahlmenüs oder Anwendungsschalter, erst wieder "mühselig" in temporäre Variablen übertragen werden müssen. Diese gestatten ihrerseits aufgrund ihres skalaren Charakters keine Generierung komplexer, temporärer Objektstrukturen, die den zuvor (vgl. Kapitel 7.2.1) beschriebenen Anforderungen genügen könnten.

Abb. 7-2 zeigt die drei Realisierungsalternativen von Verteilung und Integration im Zusammenhang. Während die Kommunikationsteile 1a) und 1b) zeitlich getrennt ablaufen, operieren 3a) und 3b) "gleichzeitig" im Sinne des Client-Server-Konzepts.

7.2.3. Entwicklungs- und Wartungs-Produktivität

Aufgrund der Dynamik des Informationsbedarfs an Management-Arbeitsplätzen kommt dem Zeitaufwand und den Knowhow-Anforderungen bei Entwicklung und Wartung rechnergestützter Anwendungen entscheidende Bedeutung zu:

- Die Knowhow-Anforderungen üben insofern globalen Einfluß auf den Zeitaufwand aus, als ein niedriges Niveau es erlaubt, daß Änderungen und Erweiterungen unmittelbar von Fachspezialisten durchgeführt werden können. Dadurch entfällt nicht nur der Zeitverlust durch die Formulierung eines Wartungsauftrags an die DV-Abteilung, es können auch Kommunikationsfehler zwischen Fach- und EDV-Abteilung vermieden werden.

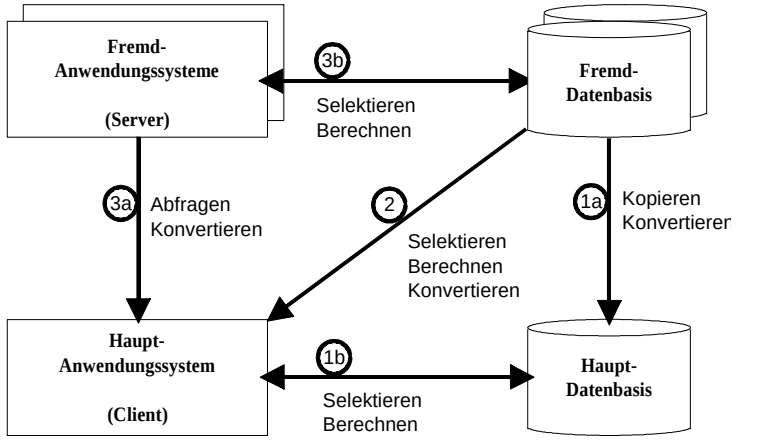


Abb. 7-2: Realisierungsalternativen der Integration verteilter Informationsquellen

❒ Unabhängig davon hängt der Zeitaufwand bei *Erweiterungen* bestehender Anwendungen davon ab, inwieweit auf vorhandene Anwendungsteile zurückgegriffen werden kann, die im Sinne einer weiteren Spezialisierung lediglich geändert oder erweitert werden können anstatt völlig neu erzeugt werden zu müssen.

In allen angebotenen Werkzeugen ist dies durch eine COPY-Funktion bzw. das Abspeichern vorhandener Anwendungsteile unter einem neuen Namen möglich. Danach können Elemente wie Importe von oder Verweise auf Informationsquellen, Dialog-Buttons samt auszuführender Aktionen, Text- oder Grafik-Views usw. gelöscht, geändert oder hinzugefügt werden.

Von diesen Struktur-ändernden Erweiterungen zu unterscheiden sind Anpassungen, die lediglich in dem Hinzufügen oder Löschen einer weiteren Ausprägung ("Instanz") eines Anwendungsteils bestehen. Ist bspw. für jedes Produkt eine Ansicht mit Monatsumsätzen realisiert, so ist die Umsetzung des Hinzukommens eines weiteren Produktes durch o.g. Kopier-Mechanismus - wie bei CommanderEIS - inakzeptabel. Datengetrieben arbeitende Werkzeuge wie bspw. Pilot oder LightShip können diesen Aufwand praktisch auf Null reduzieren, indem alle Verweise auf Informationsquellen durch aus der Datenbasis dynamisch zur Laufzeit geladene Variablen ersetzt werden. Dies erhöht zwar den anfänglichen Entwicklungsaufwand samt Entwickler-Knowhow nicht unerheblich, wird jedoch angesichts der zu erwartenden hohen Wartungsintensität schnell amortisiert. Unverzichtbar in diesem Zusammenhang ist, daß auch Drill-Down-Funktionen, z.B. zum Abruf von Detailinformationen zu einzelnen, hier insbesondere neu hinzugekommenen, Produkten ebenfalls von diesem datengetriebenen Mechanismus angepaßt werden, d.h. aus den eingelesenen Informationen abgeleitet werden.

In allen solchermaßen datengetrieben arbeitenden Werkzeugen fehlt jedoch eine echte "Instanziierung". Es existiert immer nur *ein* Fenster, dessen Inhalt bei jedem Produktwechsel neu gefüllt wird. Dies gilt auch für den EIS-Generator Forest&Trees, da auch hier keine Instanzen von VIEW-Elementen erzeugt werden können, sondern allenfalls die Belegung des Views über Variablen substituiert wird. Somit erfordert eine Erweiterungs-Anforderung der Art, daß bspw. immer zwei Produktfenster parallel angezeigt werden können, wieder das Zurückgreifen auf den bereits beschriebenen Kopier-Mechanismus. Bei Werkzeugen wie LightShip, Pilot oder CommanderEIS, die grundsätzlich nur mit einem Bildschirmfenster operieren, lassen sich derartige Anforderungen nur durch entsprechende Elementerweiterungen innerhalb der Fenster realisieren, wodurch bereits rein platzmäßig natürliche Grenzen gesetzt sind.

Erweiterungen struktureller Art in der Datenbasis, z.B. das Hinzufügen einer weiteren Spalte für eine weitere Region, können in CommanderEIS zwar durch automatisch vom Werkzeug aktivierte Rollbalken umgesetzt werden, die notwendige Anpassung des Drill-Down bleibt jedoch wiederum datengetrieben arbeitenden Werkzeugen vorbehalten. Erstreckt sich die Strukturänderung zusätzlich auf den Datentyp, d.h. soll bspw. zusätzlich zu den Monatsumsätzen je Produkt in einer weiteren Spalte noch der Name des zuständigen Vertreters, womöglich incl. einer Drill-Down-Möglichkeit bezüglich weiterer Personendaten, realisiert werden, sind auch diese Systeme überfordert.

❒ Von diesen beschriebenen Fällen ist der Zeitaufwand für *Änderungen* zu unterscheiden, die keine strukturellen Auswirkungen aufweisen, sondern lediglich die wertmäßige Belegung von Elementen in Anwendungsteilen betreffen. Beispiele hierfür sind etwa Texte von Bildschirm- oder Spaltenüberschriften, Unternehmenslogos oder generelle Darstellungsattribute wie Form und Farben von Schaltflächen, Bildschirm-, Text- oder Grafikvorder- bzw. -hintergrund. Sie weisen aufgrund der software-ergonomischen Erfordernisse in der Regel sowohl globalen als auch zugleich benutzerindividuellen Charakter auf. Unter dem Gesichtspunkt des Entwicklungs- und Wartungsaufwandes sollten diese Attribute denn auch global spezifiziert bzw. geändert werden können.

Zwar können in allen Werkzeugen (farbliche) Darstellungsattribute mehr oder weniger komfortabel global geändert werden, dies betrifft jedoch nur die Grundeinstellungen des gesamten Systems. Sollen einzelne Anwendungsteile darüber hinaus individuell gestaltet werden, so ist bei CommanderEIS das "Angreifen" jeder einzelnen Ausprägung er-

forderlich. Bei datengetriebenen Systemen wie LightShip, Pilot oder Forest&Trees reduziert sich dieser Aufwand jeweils auf das entsprechende Anwendungsteil. Aber auch hier ist es nicht möglich, einzelne Elemente von Anwendungsteilen, z.B. eine Grafikspezifikation, die in mehreren Anwendungsteilen verwendet wird, global zu ändern.

Günstiger gestaltet sich das Bild beim Inhalt statischer Bildschirm-Elemente, z.B. Überschriften oder Unternehmenslogos, die in allen Fällen als Link dynamisch "importiert" werden können. Im Falle von CommanderEIS ist hierzu der Verweis auf ein entsprechendes Text- oder Bitmap-Dokument nötig, bei LightShip können globale Variablen bzw. IMAGE-Files verwendet werden, bei Forest&Trees ist das Gestaltungsspektrum systemseitig auf Variablen für standardisierte View-Fenster-Überschriften beschränkt.

Zusammenfassend kann festgestellt werden, daß die beschriebenen Schwachstellen im wesentlichen darauf zurückzuführen sind, daß keines der Werkzeuge in der Lage ist, Informationen als *Objekte* zu behandeln, die neben ihren Datenwerten und -typen über weitere konfigurierbare und redundanzfrei komponierte Attribute und Funktionen verfügen können. Vielmehr müssen diese Eigenschaften mehr oder weniger aufwendig erst im Rahmen der Anwendungsentwicklung individuell hinzugefügt werden.

Eine deutliche Verbesserung könnte erzielt werden, wenn diese Eigenschaften den abzurufenden bzw. darzustellenden Informationen bereits auf der Ebene der Datenbasis "zugeordnet" werden könnten, und zwar unabhängig von einer konkreten Anwendung. Die Anwendungsspezifikation könnte sich dann auf das Zusammenstellen einiger weniger "Ausgangs"-Informations-Objekte beschränken. Das Wissen über ihre jeweilige Präsentationsform sowie über erlaubte bzw. mögliche Dialogfunktionen, z.B. Drill-Down, könnten sie dann "datengetrieben" quasi selbst schrittweise mitbringen.

Einzelne Anwendungsspezifikationen bestehen dann nicht mehr aus allen möglichen Anwendungsteilen, in denen sich der Benutzer nur selektiv bewegen kann, sondern nur noch aus "Einstiegsstellen", von denen aus der Benutzer individuell seine Anwendung "generativ" aufbaut. Wird eine Möglichkeit geschaffen, solche generierte Zustände abzuspeichern, wird praktisch die Anwendungsentwicklung auf Seiten der Entwickler auf die Spezifikation einer Konfigurationsmenge reduziert, und der (große) Rest ohne spürbaren Mehraufwand in die Hände des Endanwenders übertragen.

Zur Realisierung dieses Ansatzes sind allerdings grundlegende, strukturelle, methodische und konzeptionelle Änderungen der Entwicklungswerkzeuge zur Gestaltung rechnergestützter Management-Arbeitsplätze notwendig. Dabei kann auf eine Reihe innovativer Methoden

und Techniken der Informatik zurückgegriffen werden. Deren grundlegenden Eigenschaften sowie ihr potentieller Beitrag zur Realisierung des skizzierten Ansatzes sind Gegenstand des folgenden Kapitels 8. Im anschließenden Kapitel 9 wird der Versuch unternommen, den Ansatz zu einem integrativen Konzept auszuformulieren sowie anhand eines Prototyps zu realisieren. Dabei wird weitestmöglich auf bestehende Softwarewerkzeuge zurückgegriffen.

7.3. DSS-Generatoren

7.3.1. Struktur und Funktionalität

Betriebswirtschaftliche Problemstellungen weisen in der Regel eine heterogene, mehrdimensionale Struktur auf. Dies spiegelt sich auch in dem zur Exemplifizierung der Analyse eingesetzten Fallstudienmodell wieder. Es umfaßt insgesamt die folgenden 5 Dimensionen und Ausprägungen:

- Kennzahlen (Variables): s. Teilmodelle
- Jahre: 1991, 1992, 1993, 1994, 1995
- Version: Plan, Ist, absolute und prozentuale Plan-Ist-Abweichung
- Region: Nord-Ost, Mitte, Süd-West, alle Regionen
- Risikogruppe (RisikoGrp): Unfall, Feuer, Diebstahl, Transport, alle Risiken

Die heterogene Struktur resultiert daraus, daß nicht alle Kennzahlen für alle Regionen bzw. Risikogruppen existieren. Dieser (typische) Umstand korreliert in der Praxis häufig mit einer analogen Aufgabenverteilung auf verschiedene Unternehmensbereiche (Vertrieb, Prämienkalkulation, Erfolgsrechnung). Abb. 7-3 verdeutlicht diese heterogene Mehrdimensionalität, indem jeweils gleich dimensionierte Modellteile blockweise zusammen dargestellt werden.

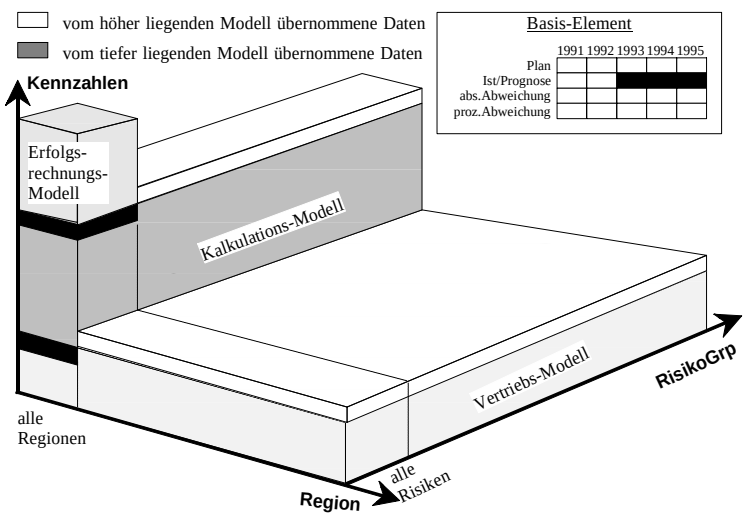


Abb. 7-3: Heterogene Mehrdimensionalität der Fallstudie

7.3.2. Verteilung und Integration

Die Umsetzung dieser Struktur mit verfügbaren konventionellen DSS-Generatoren wird wesentlich von deren strukturellen Konstrukten in Bezug auf Mehrdimensionalität beeinflußt (vgl. Kapitel 6.2.1.1). Lediglich macControl bietet dabei die Möglichkeit, Kennzahlen innerhalb ein und desselben Modells unterschiedlich zu dimensionieren. Sollen im Sinne der Speicherplatz-Effizienz "undefinierte" Modellteile vermieden werden - im Beispiel der Fallstudie entspräche dies dem Raum zur Vervollständigung des in Abb. 7-3 dargestellten Gebildes zu einem Quader - bietet sich die Aufteilung in Teilmodelle an, im Beispiel ein Vertriebs-, ein Kalkulations- und ein Erfolgsrechnungs-Modell. Diese auch aus Gründen der Modelltransparenz und Wartungseffizienz zu präferierende Lösung läßt sich prinzipiell mit den Konstrukten bspw. der Planungssprachen OneUp und CA-Compete realisieren.

In der betrieblichen Praxis - und so auch in der hier verwendeten Fallstudie - kommt jedoch erschwerend hinzu, daß diese Teilmodelle dynamisch miteinander verknüpft sind. Abb. 7-4 zeigt diese Verknüpfungen am Beispiel des Versicherungs-Controlling-Modells.

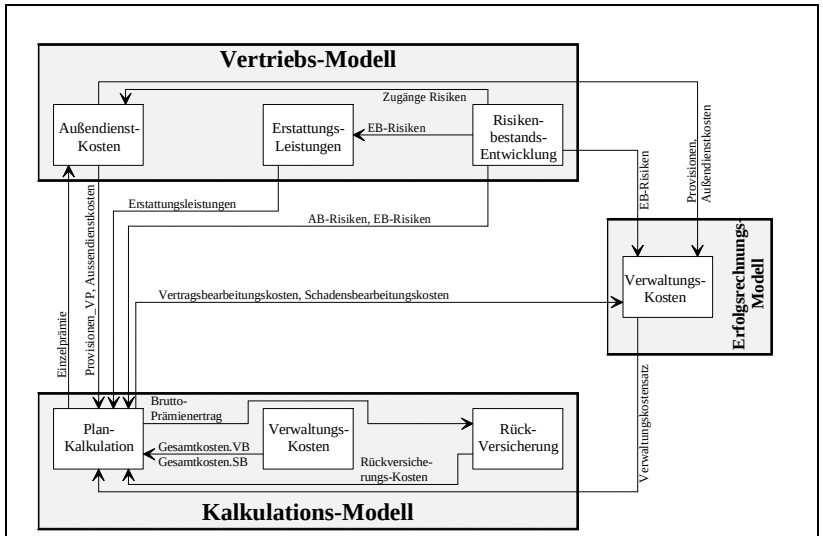


Abb. 7-4: Verknüpfungsstruktur der Fallstudie

So benötigt das Kalkulations-Modell zur Ermittlung der Prämienhöhe bspw. Informationen wie Provisionen, Außendienstkosten, Erstattungsleistungen und Vertragsbestände vom Vertriebs-Modell sowie Verwaltungskostensätze vom Erfolgsrechnungs-Modell. Andererseits erfordert die Bestimmung der Außendienstkosten im Vertriebs-Modell die Kenntnis der Prämienhöhe aus dem Kalkulations-Modell. Ähnlich sieht es beim Erfolgsrechnungs-Modell aus, das zur Bestimmung des Verwaltungskostensatzes bspw. auf Kostendaten aus dem Kalkulations- und Vertriebs-Modell angewiesen ist.

Solange es sich bei diesen Verknüpfungen - wie in der Praxis zumeist der Fall - um Bezüge auf Vorperioden handelt, die lediglich relativ selten und zu wohldefinierten Stichzeitpunkten, wie im Falle der Prämienkalkulation z.B. jährlich aktualisiert werden müssen, genügt eine lose Kopplung der Teilmodelle im Sinne eines kontrollierten Nebeneinander. Nur zu den Stichzeitpunkten werden dann die Schnittstellendaten untereinander ausgetauscht.

Handelt es sich jedoch um simultane Modellverknüpfungen, wie z.B. zwischen Erfolgsrechnungs- und Kalkulations-Modell im Falle des Verwaltungskostensatzes, ist eine Kopplung derart notwendig, daß beide Modelle gleichzeitig aktiv sind und "zirkulär" Daten austau-

schen. Diese Anforderung der Auflösung simultaner Gleichungsbeziehungen zwischen Modellen läßt sich nur mit DSS-Generatoren wie z.B. CA-Compete realisieren, die unter einer Multitasking-fähigen Betriebssystemumgebung wie MS-Windows operieren und ein entsprechendes Prozeß-Kommunikations-Protokoll wie bspw. DDE unterstützen.

7.3.3. Entwicklungs- und Wartungs-Produktivität

Die Entwicklungs- und Wartungs-Produktivität betrifft das Konstrukt-Instrumentarium in zweierlei Hinsicht:

- Zum einen die Spezifikation der Modellbeziehungen und
- zum anderen die Datenversorgung.

7.3.3.1. Datenversorgung

Bei der Datenversorgung genügt in der Regel wieder eine Stichtagsbezogene Aktualisierung von "Eingabedaten". Hierzu ist konstrukt-mäßig eine Differenzierung des DSS-Generators zwischen berechneten und nicht-berechneten Modellfeldern notwendig. Dies ist auch im Falle manueller Änderungen Produktivitäts-förderlich. Im Falle von OneUp steht hierzu bspw. ein spezielles, selektives Sichten-Kommando zur Verfügung. Im Falle von CA-Compete werden Eingabedaten zwar visualisiert, können jedoch nicht automatisch selektiert werden, so daß manuell entsprechende Sichten vom Entwickler spezifiziert werden müssen.

In beiden Fällen ergeben sich Probleme im Zusammenhang mit inhomogenen Strukturen. Sind einzugebende Kennzahlen bezüglich einer oder mehrerer Modelldimensionen auf Teilmengen der zugehörigen Ausprägungen beschränkt, wie z.B. beim anfänglichen Vertragsbestand je Versicherungssparte und Region auf das erste Modelljahr 1991, so wird dies von der Selektionsfunktion nicht berücksichtigt. Während im Falle von OneUp keinerlei Hinweis für den Anwender geliefert wird, bietet CA-Compete wenigstens visuelle Hinweise. Dafür ergeben sich dort massive Probleme, wenn (aktualisierte) Eingabedaten nachgeladen werden sollen, z.B. aus einer dBASE-Datei. Die Ladefunktion kann zwar selektiv für einzelne Kennzahlen und Dimensionen ausgeführt werden und auch optional auf das Überschreiben von originären Eingabedaten beschränkt werden, dies setzt allerdings voraus, daß jeweils alle Ausprägungen der Dimensionen, im Beispiel also alle Jahre, vom Charakter "Eingabe" sind. Als Ausweg bleibt hier nur die Aufteilung der Eingabedateien.

Eine Alternative dazu stellt wieder die dynamische Verknüpfung der Eingabedaten mit externen Anwendungssystemen im Sinne verteilter Systeme dar, z.B. in Form von Datenbank-

abfragen per SQL. Diese Lösung vermeidet die o.g. Probleme, erfordert jedoch u.U. erheblichen, wenn auch nur einmaligen Mehraufwand bei der Modellentwicklung. Die Verknüpfungen können nämlich nur in begrenztem Maße "global" definiert werden. Konkret muß es sich für eine Globalisierung um homogene und zusammenhängende Modellteile handeln. Dem kann durch "geschickten" Modellaufbau Rechnung getragen werden, erfordert jedoch entsprechend tief gehende Kenntnisse des Modellentwicklers über die Funktionalität und Rahmenbedingungen der betreffenden Konstrukte des DSS-Generators, hier der EXTERN-Funktion im Zusammenhang mit der Globalisierung. Diese Kenntnisse sind jedoch bei allen DSS-Generatoren weder aus den Handbüchern oder der Online-Hilfe zu entnehmen, noch gehen angebotene Schulungsprogramme über sporadische Einzelfallbetrachtungen hinaus. Die Bedeutung dieser Schwachstelle wird in der folgenden Betrachtung der Produktivitäts-Potentiale bei der Modellspezifikation noch eklatanter offensichtlich und führt anschließend zu Konzepten einer wissensbasierten Unterstützung von Modellbildung und -validierung.

7.3.3.2. Modellspezifikation und -validierung

Die Spezifikation der Logik mehrdimensionaler Modelle ist in starkem Maße von sich ähnelnden Beziehungen und Fallunterscheidungen geprägt. Die folgenden Tabellen zeigen diesen Zusammenhang je Kennzahl für die 3 Untermodelle des Teilmodells Vertrieb der verwendeten Beispiel-Fallstudie. Sie dokumentieren auch den typischerweise hohen Anteil eingelesener Daten aus der operativen Datenbasis sowie dessen mehrdimensionale Heterogenität. Die Tabellen sind zu diesem Zwecke in drei durch Doppelstriche getrennte Bereiche unterteilt. Der erste enthält auf nicht-konsolidierter, der zweite auf konsolidierter Ebene aus anderen Modellen oder Datenbasen einzulesende Daten. Der dritte Block schließlich umfaßt ausschließlich innerhalb des Modells neu generierte und berechnete Daten.

a) Vertriebs-Untermodell: Risikenbestands-Entwicklung

Kennzahl	Beziehung/Datenherkunft (AB=Anfangsbestand; EB=Endbestand; INT=Funktion)
Akquisitionspotential	Einlesen
AB_Risiken	1991: Einlesen sonst: EB_Risiken des Vorjahres
Akquisitionsrate	alle Regionen: Einlesen sonst: alle Regionen
Stornorate	alle Regionen: Einlesen sonst: alle Regionen
Stornierungen	INT(AB_Risiken * Stornorate)

Zugaenge_Risiken	INT(Akquisitions-Potential * Akquisitionsrate)
EB_Risiken	AB_Risiken + Zugaenge_Risiken - Stornierungen

b) Vertriebs-Untermodell: Erstattungsleistungen

Kennzahl	Beziehung/Datenherkunft (Dschn=Durchschnittl.; EB=Endbestand)
Schadensquote	Einlesen
Dschn_Schadenshoehe	1991: Einlesen sonst: Dschn_Schadenshoehe des Vorjahres * Preisindex
Preisindex	alle Regionen: Einlesen sonst: = alle Regionen
Anzahl_Schaeden	INT(EB_Risiken * Schadensquote)
Erstattungsleistungen	Anzahl_Schaeden * Dschn_Schadenshoehe

c) Vertriebs-Untermodell: Außendienstkosten

Kennzahl	Beziehung/Datenherkunft (AD=Außendienst; Dschn=Durchschnittl.)
Anzahl-Vermittler	Einlesen
sonst_Kosten-Rate	Einlesen
Anzahl_Mitarbeiter_AD	Einlesen
Sachkosten_AD	Einlesen
AB_Risiken	aus Untermodell Risikenbestands-Entwicklung
Abschluß-Provisionssatz	alle Regionen: Einlesen sonst: alle Regionen
Betreuungs-Provisionssatz	alle Regionen: Einlesen sonst: alle Regionen
Dschn_Gehalt_AD	alle Regionen: Einlesen sonst: alle Regionen
Einzelpraemie	alle Regionen: aus Kalkulations-Modell sonst: alle Regionen
Abschluß-Provision	Dschn_Abschluss-Provision * Zugaenge_Risiken
Brutto-Praemienertrag	Einzelpraemie * AB_Risiken
Dschn_Praemie/Vermittler	alle Regionen: Brutto-Praemienertrag / Anzahl_Vermittler sonst: alle Regionen
Dschn_Abschluss-Provision	Einzelpraemie * Abschluß-Provisionssatz

Betreuungs-Provision	Betreuungs-Provisionssatz * Dschn_Praemie/Vermittler * Anzahl_Vermittler
sonst_Vermittlungskosten	Zwischensumme_Provisionen * sonst_Kosten-Rate
Zwischensumme_Provisionen	Abschluß-Provision + Betreuungs-Provision
Provisionen	Zwischensumme_Provisionen + sonst_Vermittlungskosten
Personalkosten_AD	Dschn_Gehalt_AD * Anzahl_Mitarbeiter_AD+ sonst_Personalkosten_AD
sonst_Personalkosten_AD	Dschn_Gehalt_AD * Anzahl_Mitarbeiter_AD * 2/100
AD_Kosten	Personalkosten_AD + Sachkosten_AD

So ist bspw. die Ermittlung des Endbestands an Verträgen (EB_Risiken) aus Anfangsbestand (AB_Risiken), Zugängen (Zugaenge_Risiken) und Abgängen (Stornierungen) für alle Versicherungssparten (RisikoGrp), Regionen und Jahre identisch. Diesem Charakteristikum wird in Planungssprachen wie macControl, OneUp oder CA-Compete durch das Konstrukt der Globalisierung von Formeln Rechnung getragen. Im Gegensatz zu den weit verbreiteten Spreadsheet-Paketen, wie z.B. MS-Excel, werden gleichartige Beziehungen nicht durch physisches Kopieren abgebildet, sondern durch Erweiterung des Geltungsbereiches einer einzelnen Formel, wie im Falle von macControl oder CA-Compete, oder durch Einschränkungen einer ebenfalls einzelnen, aber von vorneherein für alle möglichen Kombinationen gültigen Formel, wie im Falle von OneUp. Somit muß nicht nur eine einzige Formel für die im angeführten Beispiel bereits 60 Modellzellen bei der Modellentwicklung spezifiziert werden, bei Änderungen der Modell-Logik ist auch nur eine einzige Stelle betroffen, sofern sich die Änderung ausschließlich auf denselben Geltungsbereich beschränkt, z.B. also eine Differenzierung der Zu- und Abgangsgrößen vorgenommen werden soll.

Noch gravierender stellt sich der Effizienzvorteil des Globalisierungs-Konstrukts im Falle der in der betrieblichen Praxis elementaren Konsolidierung dar. So erstreckt sich die Beziehung "Summe über alle Versicherungsarten" für die Ausprägung "alleRisiken" nicht nur auf alle Regionen, Jahre und Versionen, sondern darüber hinaus auf eine Vielzahl von Kennzahlen. Im Teilmodell Vertrieb der Beispiel-Fallstudie sind dies allein 22 von 31 betriebswirtschaftlichen Größen. Die Handhabung dieses typischen Tatbestandes wird je DSS-Generator mit unterschiedlichen Konstrukt-Strategien unterstützt, deren Vor- und Nachteile im folgenden detailliert untersucht werden sollen:

❑ *Variablentyp Konsolidierung bei OneUp*

Im Falle von OneUp reduziert sich die Spezifikation der insgesamt immerhin 1320 Konsolidierungs-Beziehungen auf *eine* Definition der hierarchischen Beziehung zwischen "alleRisiken" und den zugehörigen Versicherungsarten. Da alle Kennzahlen (Standarddimension *Variables*) automatisch den Variablentyp *Consolidated* besitzen, ist umgekehrt lediglich bei den verbleibenden 9 Variablen der Typ auf *Detail* umzusetzen.

Nachteile dieses Konzepts ergeben sich erst dann, wenn die zu globalisierende Beziehung nicht vom Typ Konsolidierung ist bzw. nicht für alle Modelldimensionen (außer den Basisdimensionen) vom Typ Konsolidierung ist. In der Praxis trifft dies typischerweise für das Problem der Kostenaufteilung zu. Sollen bspw. die allgemeinen Verwaltungskosten entsprechend dem jeweiligen Bestand an Versicherungsverträgen auf Regionen und Versicherungsarten aufgeteilt und dann wieder je Region und Versicherungsart konsolidiert werden, müssen individuelle Konsolidierungs-Beziehungen für beide Dimensionen spezifiziert werden. Dies kann allerdings unter Wartungsgesichtspunkten insoweit dynamisch gehalten werden, als die Fallunterscheidung zwischen Konsolidierung und Kostenaufteilung sich auf die Abfrage des Typs der Dimensionsausprägung beschränkt. Neu hinzukommende Risikogruppen oder Regionen erfordern ebensowenig Anpassungsoperationen in der Modell-Logik wie mehrstufige Konsolidierungs-Hierarchien.

❑ *Formel-Globalisierung in CA-Compete und macControl*

DSS-Generatoren ohne generische Konsolidierungs-Mechanismen, wie z.B. CA-Compete und macControl, verfolgen das umgekehrte Prinzip der Formel-Globalisierung auf Teilbereiche des mehrdimensionalen Modellraums. Die Funktionalität von macControl geht dabei insofern über die von OneUp hinaus, als es bei nur unwesentlichem Mehraufwand in puncto Modellspezifikation die Nachteile in Form der Beschränkung auf Konsolidierungs-Beziehungen vermeidet. Der Mehraufwand liegt darin, daß für jede Kombination von *Zuordnungen*, die hier u.a. der Abbildung der Konsolidierungs-Hierarchien dienen, eine eigene Konsolidierungs-Formel definiert werden muß, da keine Zuordnungen über Zuordnungen möglich ist. Das Zuordnungs-Konstrukt ersetzt dabei nicht nur vollständig das Variablentyp-Konstrukt, es gestattet dem Entwickler insbesondere die Spezifikation beliebiger, eigener "Dimensionstypen", die eben nicht nur auf die Dimension Variablen beschränkt sind.

Im Falle von CA-Compete existiert kein vergleichbares Konstrukt für die Bildung von Ausprägungs-Teilmengen je Dimension. Stattdessen findet eine Unterscheidung zwischen Gültigkeits- und Anwendungs-Bereich von (globalen) Formeln statt. Somit ergeben sich prinzipiell zwei Möglichkeiten für die Spezifikation und Anwendung globaler Formeln:

❑ Zum einen können sie bei ihrer Erstellung sofort angewendet werden. Das setzt voraus, daß Anwendungsbereich und Gültigkeitsbereich der Spezifikation identisch sind. Bei der Beispiel-Fallstudie trifft dies auf die Kennzahlen Akquisitions-Rate und Storno-Rate zu, die am effizientesten jeweils für die konsolidierte Ausprägung "alleRisiken" der Dimension "RisikoGrp" sowie global für die Dimensionen "Jahre" und "Region" zu spezifizieren sind.

❑ Zum anderen können globale Formeln zunächst nur für einen bestimmten Gültigkeitsbereich spezifiziert werden. Auf diese Weise können dann prinzipiell beliebig viele gültige Formeln je Modellzelle entstehen, aus denen in einer zweiten Phase der Formel-Anwendung die relevante auszuwählen ist. Der Effizienzvorteil ergibt sich daraus, daß bei dieser Anwendung beliebige (allerdings) zwei-dimensionale Teilmengen gebildet werden können. Zur Verdeutlichung soll folgender Ausschnitt aus der Beispiel-Fallstudie dienen:

Typische Kandidaten dafür sind Konsolidierungs-Formeln, die - wie gesehen - in der Regel gleichermaßen für viele, aber nicht alle Variablen eines Modells gelten. In der Fallstudie könnte so eine erste Summen-Formel über alle Basis-Regionen für "alleRegionen" und global sonst, eine zweite über alle Basis-Risiken für "alleRisiken" und global sonst definiert werden. Schließlich könnten weitere Formeln, die insgesamt global definiert werden, die "normalen" Kennzahl-Verknüpfungen auf nicht-konsolidierten Ebenen abdecken, bspw. zur Ermittlung von Stornierungen, Risiken-Zugängen und -Endbeständen. Abb. 7-5 zeigt die drei Modellansichten mit den jeweiligen (globalen) Formeln.

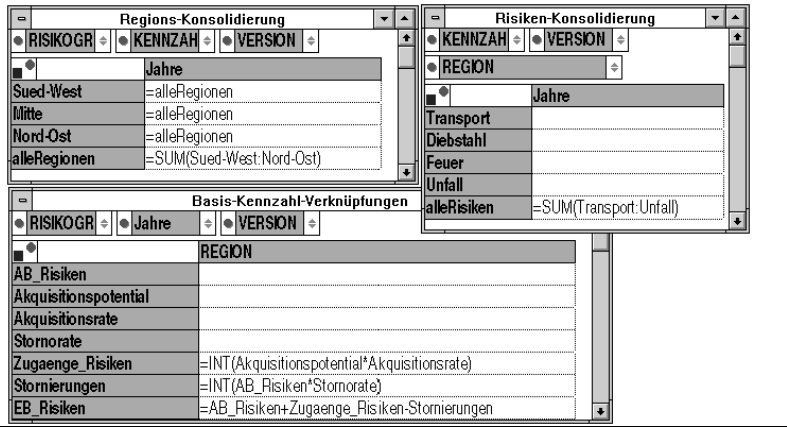


Abb. 7-5: Spezifikation globaler Formeln in CA-Compete

Bei der anschließenden Anwendung der Formeln auf Teilbereiche des Modells kann dann jeweils aus der Menge der definierten Formeln ausgewählt werden. Abb. 7-6 zeigt das Auswahl-Menü für die Festlegung aller bezüglich Region zu konsolidierenden Kennzahlen, was im Falle der zuvor beschriebenen Art der Formelspezifikation in einem einzigen Arbeitsschritt geschehen kann. Der konkrete Anwendungsbereich ergibt sich aus der (zwei-dimensionalen) Selektion der als Zeilen und Spalten gewählten Dimensionen sowie die (festen) Einstellungen für die übrigen Dimensionen. Im Beispiel sind diese global für alle Jahre und Versionen, was an den Kreissymbolen erkennbar ist. Bezüglich der Region ist (natürlich) nur die konsolidierte Ausprägung ausgewählt.

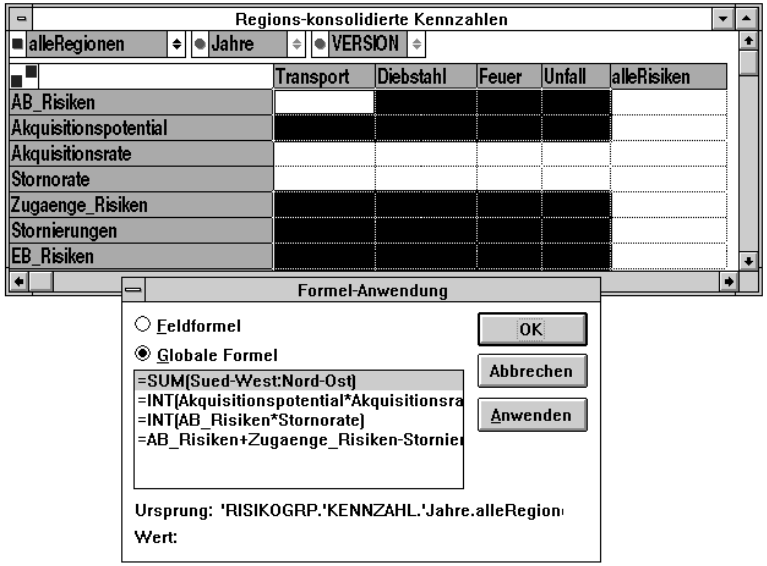


Abb. 7-6:	Anwendung globaler Formeln in CA-Compete

Ferner bietet sich damit auch die Möglichkeit, bei später im Rahmen der Modellpflege hinzu kommenden Kennzahlen per bloßer Anwendung auf diese bereits existierenden Formeln selektiv zuzugreifen.

Aufgrund umfangreicher Erfahrungen in der Praxis und beim Einsatz in zahlreichen Lehrveranstaltungen wird diese Flexibilität allerdings "erkauft" mit einer als dramatisch einzustufenden Erhöhung der Anforderungen an das "räumliche" Vorstellungs- und Abstraktions-Vermögen der Modellentwickler. Probleme ergeben sich insbesondere in Bezug auf die Vollständigkeit und Validität der Modellspezifikation.

So werden bei OneUp prinzipiell bis zu drei - bei Fehlern in der Spezifikation von Fallunterscheidungen sogar mehr - Beziehungen für eine Modellzelle spezifiziert (Variablenregel, Periodenregel und Konsolidierungsregel). Welche im konkreten Einzelfall der Rechenalgorithmus des DSS-Generators anwendet, wird durch eine starr vorgegebene Prioritätssteuerung bestimmt, deren Wirkung im Zusammenhang mit den standardmäßig globalen Regeln vom Entwickler einkalkuliert werden muß. So wird bspw. im Falle einer summarischen Zeitausprägung (z.B. Summe der Quartale eines Jahres) konstrukt-bedingt eine manuell zu spezifizierende Fallunterscheidung (Ausschluß der Summation) für alle Quotienten-Kennzahlen des Modells fällig. Der hierfür geschaffene Variablentyp *Ratio* bezieht sich nämlich nur auf Konsolidierungs-Dimensionen, und die Zeit-Dimension ist keine solche.

Bei CA-Compete ergibt sich darüber hinaus Entscheidungsbedarf in Bezug auf die Allokation globaler Formeln im n-dimensionalen Raum sowie die Konfiguration Anwendungsoptimaler Modell-Ansichten. Einen Eindruck über die sich dabei ergebende Komplexität bereits kleiner, betriebswirtschaftlich einfacher Modelle gibt die folgende Abb. 7-7.

Darin sind (nur) für das Untermodell "Risiken-Bestand-Entwicklung" der Beispiel-Fallstudie optimale globale Formeln und deren Allokation (Spalte: Definieren in) sowie optimale Ansichts-Konfigurationen für deren Anwendung (Spalte: Anwenden auf) dargestellt. Dabei sind Aggregationen nach gleich zu behandelnden Fällen vorgenommen worden (Spalte: (Unter-)Typ). Diese decken sich teilweise mit den Variablentypen des DSS-Generators OneUp, erweitern aber dessen Spektrum um weitere Fälle. Diese können als generische Beziehungstypen für betriebswirtschaftliche Modell-Anwendungen angesehen werden.

7.4. Wissensbasierte Modellbildung

7.4.1. Konzept und Unterstützungsphasen

Da die Entwicklung eines derartigen Anweisungs-Katalogs für die Modellbildung nur mit tiefgehenden Kenntnissen über die Konstrukte von DSS-Generatoren, deren Zusammenwirken und Folgewirkungen für die spätere Modellpflege möglich ist, andererseits diese zwar nirgendwo zugänglich dokumentierten, aber immanent existenten "Regeln" gehorcht, muß es

Legende: Dimensions-Anordnung: feste Dimension, vertikal, horizontal
Dimensions-Einstellung(en): G global, K Konsolidierungs-Objekt, B Basis-Objekte (nicht-konsolidiert)

Formel			Definieren in:					Anwenden auf:				
(Unter-) Typ		Formel-Beziehung	Jahre	Version	Kennzahl	Risiko Grp	Region	Jahre	Version	Kennzahl	Risiko Grp	Region
Konsolidierungen		SUM RisikoGrp B	G	G	K	K	G	G	G	AB_Risiken Akq.Potential Stornierungen Zugaenge_Ris. EB_Risiken	K	G
		SUM Region B	G	G	G	G	K	G	G	AB_Risiken Akq.Potential Stornierungen Zugaenge_Ris. EB_Risiken	B	K
Quotient	kons.Eingabe	Zugaenge_Ris. / Akq.Potential	G	G	Akq.Rate	K	G	G	G	Akq.Rate	K	G
		Stornierungen / AB_Risiken	G	G	StornoRate	K	G	G	G	StornoRate	K	G
Referenz	abs.	=Region K	G	G	G	G	B	G	G	Akq.Rate StornoRate	B	B
	relativ	=EB_Risiken.1991	1992	G	AB_Risiken	G	G	1992	G	AB_Risiken	B	B
		=EB_Risiken.1992	1993	G	AB_Risiken	G	G	1993	G	AB_Risiken	B	B
		=EB_Risiken.1993	1994	G	AB_Risiken	G	G	1994	G	AB_Risiken	B	B
		=EB_Risiken.1994	1995	G	AB_Risiken	G	G	1995	G	AB_Risiken	B	B
Objekt-Verknüpfung	ein-dimensional	INT (AB_Risiken *StornoRate)	G	G	Stornierungen	G	G	G	G	Stornierungen	B	B
		INT (Akq.Potent. *Akq.Rate)	G	G	Zugaenge Risiken	G	G	G	G	Zugaenge Risiken	B	B
		AB_Risiken +Zugaenge_Ris. -Stornierungen	G	G	EB Risiken	G	G	G	G	EB Risiken	B	B

Abb. 7-7: Anweisungs-Katalog für optimale Modellspezifikation in CA-Compete

möglich sein, ein Unterstützungssystem zu entwickeln, daß diese "Optimierungs"-Aufgaben übernehmen kann. In letzter Konsequenz kann dieses im folgenden prototypisch anskizzierte Unterstützungssystem in eine automatisierte Generierung der Modellspezifikation münden. Damit wäre eine Lösung gefunden, die beide eingangs aufgestellten Anforderungen erfüllt:

- Einerseits gewährleistet sie die sichere und verläßliche Erstellung vollständiger und valider Modelle durch nicht nur mathematisch überdurchschnittlich begabte bzw. geschulte Mitarbeiter von betriebswirtschaftlichen Fachabteilungen,

- andererseits erzwingt sie nicht den Verzicht auf die Nutzung der Effizienzvorteile leistungsfähiger Konstrukte moderner DSS-Generatoren, wie z.B. globaler Spezifikationen, bei der Modellpflege.

Ein rechnergestütztes Werkzeug zur Unterstützung dieser Anforderungen muß folgende Phasen abdecken:

- Erfassung von Modellstruktur und -beziehungen in einer für Fachspezialisten leicht bedienbaren, transparenten Form, unabhängig vom später für die Implementierung verwendeten DSS-Generator.
- Überprüfung der Modellbeziehungen auf Vollständigkeit und Widerspruchsfreiheit.
- Komprimierung bzw. Reduktion der (redundanten) Modellspezifikation durch Ableitung strukturell gleichartiger Beziehungstypen und Zuordnung der Benutzerspezifikationen zu diesen.
- Auswahl des für die Implementierung optimal geeigneten DSS-Generators unter Berücksichtigung der für die zuvor abgeleiteten Beziehungstypen und deren Ausprägungen effizientesten Konstrukte.
- Generierung einer "Anweisungsliste" für die optimale Implementierung des Modells im Hinblick auf die verfügbaren Konstrukte des ausgewählten DSS-Generators bzw. automatische Generierung der Modellspezifikation in dem DSS-Generator.

7.4.2. Konsistente Erfassung der Modellspezifikation

Für die Erfassung der Modellspezifikation durch Fachspezialisten erweisen sich - wie zuvor gesehen - die für die spätere Implementierung und Wartung zwar effizienten, komplexen Konstrukte von DSS-Generatoren als ungeeignet. In dieser Entwurfsphase ist jedoch eine Optimierung unter Implementierungsgesichtspunkten noch gar nicht notwendig, im Gegenteil sogar hinderlich. Stattdessen ist eine in dieser Phase durchaus redundante Spezifikation auf elementarer Ebene vorzuziehen, da sie dem Vorstellungsvermögen von Fachspezialisten näher kommt, und dadurch dessen "Entscheidungsfähigkeit" darüber, wie die Modellbeziehungen im einzelnen auszusehen haben, verbessert. Mit elementarer Ebene sind dabei einzelne Modellzellen oder sachlogisch, d.h. in betriebswirtschaftlichen Termini zusammengehörende Modellteile zu verstehen, z.B. das Akquisitionspotential für die konsolidierten Regions- und RisikoGrp-Ausprägungen je Jahr und Version.

Aufgrund dieser elementaren Ebene ist auch leicht eine visualisierte Vollständigkeitskontrolle in Analogie zum Exception-Reporting von EIS zu integrieren. Die Widerspruchsfreiheit der Modellspezifikation kann dadurch quasi "erzwungen" werden, daß die Spezifikation in zwei Schritte aufgeteilt wird. Im ersten Schritt wird durch Fallunterscheidungen eine disjunkte Aufteilung des Spezifikationsraumes vorgenommen, z.B. konsolidierte und nichtkonsolidierte Ausprägungen je Dimension. Im zweiten Schritt wird dem Benutzer dann auf dieser Basis der sich ergebende Spezifikations-Raum "zum Ausfüllen" vorgegeben. Solange "Lücken", d.h. undefinierte Bereiche im Spezifikationsraum bestehen, kann der Entwickler durch entsprechende Markierungen, darauf aufmerksam gemacht werden. Evtl. identische Berechnungs-Anweisungen in einzelnen Feldern dieses Spezifikations-Raums werden bewußt im Interesse der Modelltransparenz in Kauf genommen und können dem "Entwerfer" durch Kopier-Operationen erleichtert werden. Abb. 7-8 zeigt ein Beispiel für den Prototyp eines derartigen Konsistenz-prüfenden, DSS-Generator- unabhängigen Modellierungssystems.

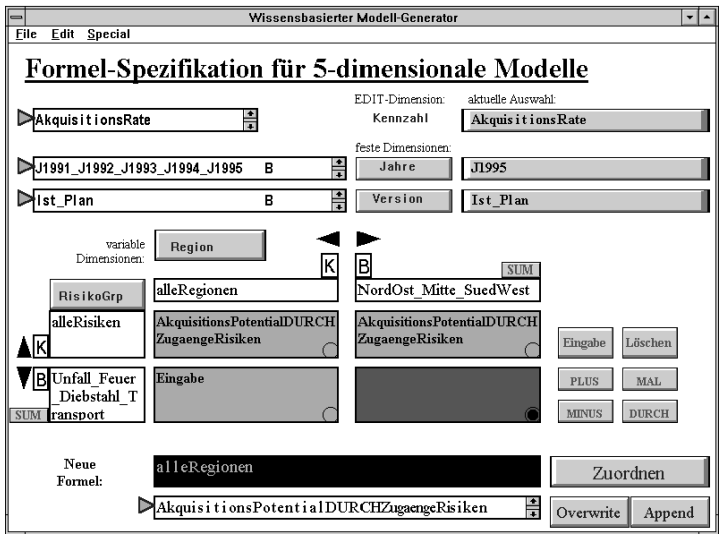


Abb. 7-8:	Konsistenzprüfende Erfassung von Modellspezifikationen

Aufgrund der benötigten "Alarm"-Funktionalitäten wurde es mit dem EIS-Generator LightShip entwickelt. Als "Datenbasis" für die Ablage bzw. den Abruf von Modellstruktur und -Spezifikationen kommen sowohl relationale Datenbanksysteme in Betracht, als auch - wie im Prototyp aufgrund der anschließenden Phasen realisiert - direkt wissensbasierte DSS-Generatoren mit objektorientierter Wissensrepräsentation. Diese zweite Alternative bietet den Vorteil, daß auch während der Erfassungsphase bereits wissensbasierte Analysen und Spezifikations-Empfehlungen resp. Erklärungen für erkannte Inkonsistenzen, z.B. insbesondere bezüglich (ungewollter) simultaner Beziehungen, integriert werden können.

Im unteren Teil kann zwischen allen Ausprägungs-Kombinationen - im Sinne von macControl wären dies Zuordnungen - von zwei in einer vorgeschalteten Phase (disjunkt) aufgeteilten Dimensionen geblättert werden. Jede Kombination visualisiert farblich den Spezifikationsstatus, z.B. grün incl. Anzeige der Beziehung für definiert und konsistent, rot für (noch) undefiniert bzw. fehlerhaft. Auch gefüllte Felder können "rot" werden, wenn bspw. Simultangleichungen von einer als Erweiterung denkbaren Konsistenz-Prüfungs-Komponente erkannt werden. Der Entwickler müßte dann aufgrund entsprechender Hinweise über die involvierte Beziehungskette bestätigen oder verwerfen, ob dies gewollt ist.

Für die eigentliche Spezifikation der Modellbeziehungen steht ein editierbares Feld zur Verfügung, in das bestehende Beziehungen, Dimensionsausprägungen und Operatoren durch Auswahl- und Kopieroperationen geladen bzw. manuell bearbeitet werden können. Danach können sie einem oder mehreren der Ausprägungskombinationen "zugeordnet" werden, wodurch deren Status unter Hinzuziehung der Konsistenz-Komponente angepaßt wird.

Im oberen Teil werden alle übrigen Dimensionen in bei Bedarf rollbaren Auswahl- und Einstellungs-Menüs angeboten, deren Vollständigkeit ebenfalls durch farbliche Ausnahme-Marker gekennzeichnet werden kann. Im Normalfall dürfte die Kennzahl-orientierte Erfassung die transparenteste sein. Optional ist jedoch auch ein Blättern durch die Dimensionen denkbar und im Erfassungssystem bereits vorbereitet, d.h. daß jede beliebige Dimension für den unteren Teil ausgewählt werden kann, z.B. Variablengruppen über Zeitgruppen. Ein direkter Bedarf dafür besteht allerdings nicht, da Redundanzen der Spezifikation, z.B. bezüglich in gleicher Weise auf konsolidierter Ebene zu summierende Variablen, in der folgenden Komprimierungsphase vom System sowieso erkannt und beseitigt werden.

7.4.3. Klassifizierende Komprimierung der Modellspezifikation

Bei der zuvor beschriebenen Form der elementaren Modellspezifikation wird eine Vielzahl von Redundanzen erzeugt, bspw. wird die konsoliderende Summation über Basisausprägun-

gen der Dimensionen Region und RisikoGrp für die Kennzahlen AB_Risiken, Akquisitions-Potential, Stornierungen, Zugaenge_Risiken und EB_Risiken anzutreffen sein. Ziel dieser zweiten Phase ist es nun, diese durch geeignete Gruppenbildung zu beseitigen und die Voraussetzung für die anschließende Phase der Generierung von Implementierungs-Empfehlungen bzw. der automatischen Implementierung selbst vorzubereiten.

Hierfür können im einfachsten Fall typische Gruppenbildungs-Operationen relationaler Datenbanksysteme eingesetzt werden. Bei der im Prototyp präferierten und realisierten Alternative wird dies durch entsprechende Methoden und Funktionen in der mit dem wissensbasierten DSS-Generator Kappa-PC implementierten Komponente übernommen. Dies bietet den Vorteil, daß nicht in jedem Fall alle Beziehungstypen neu generiert werden müssen, sondern auf einen Pool vordefinierter, aber erweiterbarer betriebswirtschaftlicher Beziehungstypen zurückgegriffen werden kann. Das Problem der Generierung von Beziehungstypen und deren Ausprägungen reduziert sich damit zu großen Teilen auf ein im Expertensystembereich bereits gut beherrschtes Klassifizierungs-Problem. Abb. 7-9 zeigt einen Ausschnitt der so generierten Objekt-Struktur.

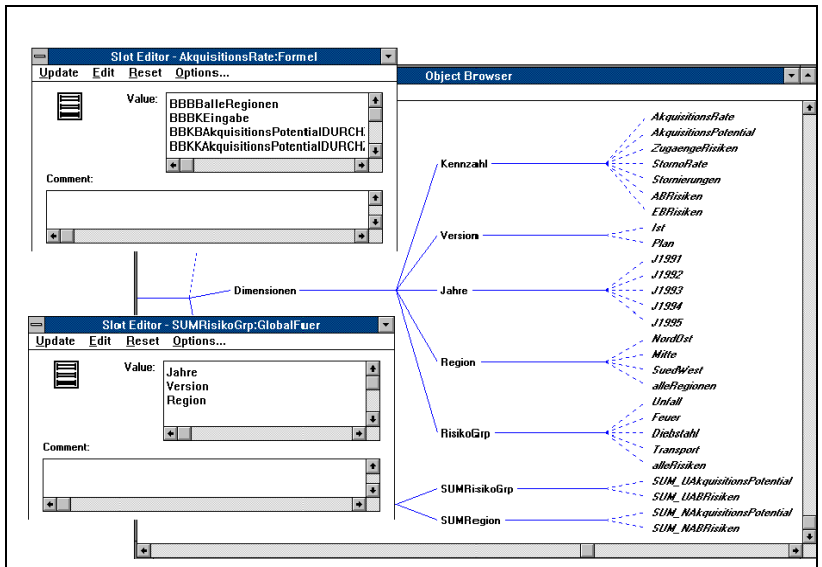


Abb. 7-9: Klassifizierende Komprimierung von Modellspezifikationen

Im oberen Teil ist die direkt aus der Phase 1 übernommene bzw. per Import-Funktion aus einer entsprechenden Datenbank geladene Spezifikations-Struktur mit einem Beispiel für die interne Objektstruktur dargestellt. Sie umfaßt alle jeweiligen Formeln einschließlich Informationen über ihre Geltungsbereiche in Form von Referenzen auf Zuordnungen.

Im unteren Teil ist die daraus abgeleitete, komprimierte, d.h. nach Beziehungstypen und Zuordnungen geordnete Struktur dargestellt. Die Objekte dieser Struktur enthalten umgekehrt in Listenform Informationen über die betroffenen Dimensionsausprägungen.

7.4.4. Konsultative Implementierung der Modellspezifikation

Im letzten Schritt kann nun unter Rückgriff auf eine Regel-orientierte Wissensbasis durch Auswertung der generierten Struktur von Beziehungsobjekten (Klasse Formeln) sowohl die Auswahl geeigneter DSS-Generatoren, die Generierung von Implementierungs-Empfehlungen als auch im Extremfall die Generierung der Implementierung selbst vorgenommen werden. Im beschriebenen Prototyp beschränkt sich diese Komponente im Moment noch auf die Generierung von Implementierungs-Empfehlungen für den DSS-Generator CA-Compete. Abb. 7-10 zeigt beispielhaft den Output für die Konsolidierungsformeln der RisikoGrp-Ausprägung "alleRisiken" bzw. Region-Ausprägung "alleRegionen".

7.5. Zusammenfassende Wertung und Ausblick

Der im vorangegangenen Kapitel 7.4 beschriebene Prototyp zeigt exemplarisch das Leistungsvermögen der Integration von Anwendungen aller zuvor untersuchten Klassen von Werkzeugen zur Entwicklung rechnerunterstützter Arbeitsplätze:

- ❑ EIS-Generatoren als integrative, leicht-bedienbare und individuell an die Erfordernisse von Benutzerklassen anpaßbare Bedieneroberflächen,
- ❑ konventionelle und wissensbasierte DSS-Generatoren zur Übernahme spezialisierter, komplexer Teilaufgaben im Sinne von im Hintergrund operierenden Dienstleistern.

Der Prototyp stellt somit selbst ein Beispiel für einen rechnerunterstützen Arbeitsplatz (RAP), den des Modellentwicklers, dar. Er macht jedoch insbesondere auch deutlich, in welch hohem Maße innovative Ansätze der Informatik nutzbringend eingesetzt werden können, speziell objektorientierte Techniken am Beispiel der Beziehungsgruppen-Bildung und deren Auswertung sowie Techniken verteilter Anwendungssysteme am Beispiel der Kooperation wissensbasierter und konventioneller Komponenten. Damit sind die Einsatzmöglichkeiten jedoch in keiner Weise in vollem Umfange ausgeschöpft.

```
Empfehlungen fuer CA-Compete-Modellierung:
-Modell: VERTRIEB.MDL

-Formel:  SUMRisikoGrp
--GLOBAL Definieren fuer:
   Kennzahl
   Jahre
   Version
   Region
--SPEZIELL definieren fuer:
   RisikoGrpK

--Anwenden auf KENNZAHL:
   AkquisitionsPotential
   ABRisiken
--in den SPEZIELLEN Dimensionen
   RisikoGrpK
--und den GLOBALEN Dimensionen
   Jahre
   Version
   Region

-Formel:  SUMRegion
--GLOBAL Definieren fuer:
   Kennzahl
   Jahre
   Version
   RisikoGrp
--SPEZIELL definieren fuer:
   RegionK

--Anwenden auf KENNZAHL:
   AkquisitionsPotential
   ABRisiken
--in den SPEZIELLEN Dimensionen
   RegionK
   RisikoGrpB
--und den GLOBALEN Dimensionen
   Jahre
   Version
```

Abb. 7-10:	Implementierungs-Empfehlung

Im folgenden dritten Teil sollen daher nach einer kurzen methodisch orientierten Beschreibung der Charakteristika dieser Auswahl innovativer Ansätze weitere Einsatzmöglichkeiten und Perspektiven für zukünftige RAP-Werkzeuge untersucht und an Prototypen exemplarisch illustriert werden.

Teil III

Integratives Werkzeug-Konzept

8. Einsatzmöglichkeiten innovativer Informatik-Ansätze

8.1. Objektorientierte Programmierung

8.1.1. Konstruktklassifizierung

Der potentielle Nutzen des Einsatzes objektorientierter Prinzipien bei der Entwicklung von DV-Anwendungen generell und (Führungs-)Informationssystemen im speziellen kann aus zwei Blickwinkeln begründet werden, die auf eine komplementäre Menge von Basiskonstrukten zurückzuführen sind:

- Aus Sicht der *Implementierung* ergeben sich Vorteile in Bezug auf inkrementelle Entwicklung, minimale Nebenwirkungen notwendig werdender Änderungen sowie effiziente Umsetzung globaler Eigenschaften und stufenweiser Spezialisierungen. Bewirkt wird dies durch die relative Unabhängigkeit einzelner Systemteile sowie den konsequenten Einsatz des Prinzips der Wiederverwendung.

- Aus Sicht der *Anwendung* ergeben sich Vorteile insbesondere durch die Möglichkeit, ein hohes Maß an Übereinstimmung von kognitivem Modell des Anwenders mit der realisierten Struktur des Anwendungssystems herstellen zu können. Einheiten in der Vorstellungswelt des Anwenders (und Auftraggebers) entsprechen direkt Elementen der Benutzeroberfläche des DV-Systems, diese wiederum korrespondieren isomorph mit vom Entwickler zu spezifizierenden Programmteilen. Dadurch entfallen im günstigsten Fall Übersetzungs- oder Zuordnungsprobleme zwischen Anwender und Entwickler in Bezug auf die Identifikation zu ergänzender bzw. betroffener und zu modifizierender Systemkomponenten im Falle von Änderungen oder Erweiterungen des Leistungsumfangs von Anwendungen; zumindest besteht bei konsequenter und konstruktgerechter Anwendung ein Potential zu ihrer Reduzierung.

Trotz zunehmend starker Durchdringung von Softwaremarkt und Anwendungsentwicklung mit dem Gedankengut der Objektorientierung hat sich bis heute keine allgemein anerkannte Theorie notwendiger Mindestanforderungen, d.h. kein Satz elementarer Konstrukte herausgebildet (vgl. [Nastansky 90, Endbenutzercomputing], S. 239). Dies muß auf die verschiedenen Informatikbereiche mit jeweils eigenen Beweggründen zurückgeführt werden, die als Geburtshelfer fungiert haben. Zu diesen historischen Wurzeln sind zumindest die Künstliche Intelligenz, die Simulation und die Graphik-Programmierung zu zählen (vgl. [Kreutzer 90,

OOP], S. 217), aus denen sich lokale oder integrative Werkzeuge entwickelt haben. Somit ist der definitorische Bereich auch stark durch eine enge Kopplung an Entwicklungswerkzeuge und Programmiersprachen geprägt. Ein typisches Beispiel hierfür mit sehr weitgehender Anforderungsdefinition für Objektorientierung kann in den "Sieben Stufen zur objektbasierten Glückseligkeit" von Meyer gesehen werden (vgl. [Meyer 90, Objektorientierte Softwareentwicklung], S. 65-68)

Als gemeinsames, abstraktes Oberziel aller Richtungen läßt sich jedoch die Suche nach Wegen zur effektiven Bewältigung der *Komplexität* von Aufgabenstellungen ableiten. Je nach dem Schwerpunkt der verfolgten Vorgehensweise, deduktiv abstrahierend oder induktiv experimentell, führt dies zu der in Abb. 8-1 dargestellten Zuordnung und Klassifizierung wesentlicher Konstrukte (vgl. [Kreutzer 90, OOP], S. 219):

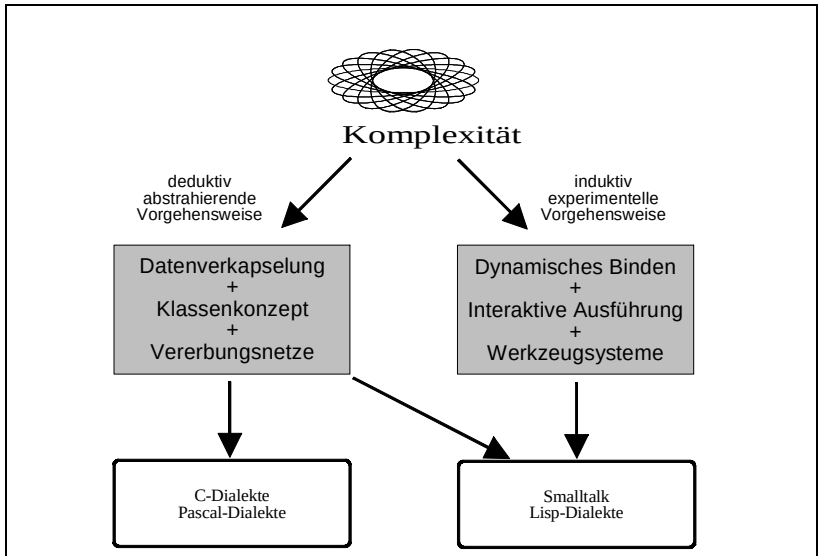


Abb. 8-1:	OOP-Konstrukt-Philosophien zur Komplexitätsreduktion (Quelle: [Kreutzer 90, OOP], S. 219)

Im Mittelpunkt des ersten Zweigs, der dem Prinzip der Problemzerlegung oder Vereinfachung folgt, steht die deduktiv abstrahierende Vorgehensweise. Als konstruktive Hilfsmittel werden Datenverkapselung, Klassenkonzept und Vererbung eingesetzt. Ihre Umsetzung findet sich an erster Stelle in objektorientierten Erweiterungen der klassischen Programmiersprachen, z.B. C und Pascal.

Der zweite Zweig, der dem Prinzip des Prototyping oder Delegation folgt, orientiert sich an der induktiv experimentellen Vorgehensweise. Wesentliche Konstrukte prozeduraler Art sind hierbei Dynamisches Binden, interaktive Ausführung und Entwicklungsumgebungen. Typische Vertreter sind bspw. Smalltalk und Lisp-Dialekte.

Beide Aspekte der Komplexitätsbewältigung kommen speziell bei Führungsinformations- bzw. -unterstützungssystemen in Betracht. Die strukturellen Aspekte decken sich mit den in Kapitel 2 dargestellten Erfordernissen der Aggregation und des Drill-Down, die dynamischen Aspekte korrelieren mit dem adhoc-Charakter und den hohen Flexibilitätsanforderungen von EIS.

Darüber hinaus impliziert die Komplexitäts-reduzierende Verwendung geschlossener, eigenständiger Elemente (Objekte) ein verändertes, dezentrales Konzept der Verarbeitungssteuerung durch Nachrichtenaustausch (message passing).

Der somit mehrdimensionale, potentielle Beitrag objektorientierter Prinzipien zur effektiveren Entwicklung, Gestaltung und Betrieb von rechnerunterstützten Systemen für den Arbeitsplatz von Führungskräften wird daher anschließend an folgenden Konstrukten dargestellt:

- *Datenkapselung/Objekt*
- *Klassen*
- *Vererbung*
- *Message Passing/Polymorphismus/Dynamisches Binden*

8.1.2. Die Konstruktbeiträge im einzelnen

8.1.2.1. Datenkapselung/Objekt

Zentrale Eigenschaft der Objektorientierung ist die Verkapselung von Daten und darauf zulässigen Operationen in Objekten, so daß Zugriffe auf und Modifikationen an den Daten ausschließlich von den zum Objekt selbst gehörenden Prozeduren (Methoden) im Rahmen des von diesen Methoden definierten Schnittstellenprotokolls erfolgen können. Entsprechend dem Prinzip des "information hiding" bleiben objekt-interne Implementierungsdetails sowohl der Datenstruktur als auch der Methoden nach außen, d.h. speziell für alle "anfragenden" bzw. "auftraggebenden" Systemkomponenten (clients), verborgen. Durch

diese Lokalität wird eine relative Implementierungsunabhängigkeit erzielt. Daneben stellt das Objekt-Konstrukt im Zusammenhang mit Informationssystemen vor allem ein effektives Modellierungs-Instrument dar, indem komplexe, heterogene Informationsobjekte in Struktur und Verhalten gesamtheitlich als Einheit abgebildet werden können.

Auf diese Weise lassen sich wesentliche Komponenten der Informationsfacetten des Beispiels in Kapitel 3 (vgl. Abb. 3-4) unmittelbar realisieren (vgl. die exemplarische Implementierung mit KappaPC in Abb. 8-2):

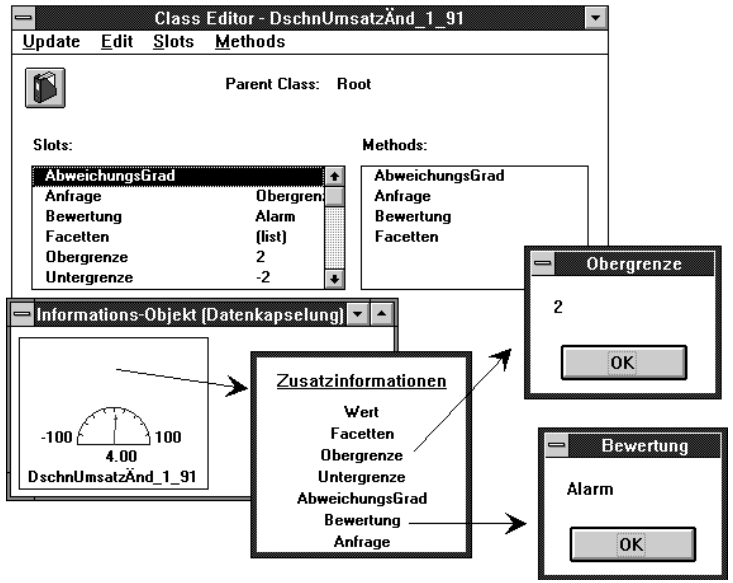


Abb. 8-2:	Datenkapselung am Beispiel eines Informationsobjektes

Slots des zu definierenden Objekts "durchschnittliche Umsatzänderung 1. Quartal 1991" nehmen neben dem Zahlenwert weitere Aspekte wie Bewertung und Grenzwerte oder auch Definitionen, Herkunft, Detaillierungsrichtungen und Kommentar auf. Zugeordnete Methoden übernehmen nicht nur die Belegung, Änderung und Anzeige dieser Zusatzinformationen, sondern sorgen auch für die bedarfsweise, dynamische Ermittlung abgeleiteter Informationen, wie z.B. Grenzwertabweichungen bezüglich Grad und Bewertung oder Trend. Die Steuerung des Angebots abrufbarer Objekt-Informationen läßt sich lokal, d.h. ohne übergeordnete, koordiniderende Instanz, realisieren. Im Beispiel der Abb. 8-2 ist dies durch ein per rechtem Maus-Klick abrufbares Popup-Menü objekt-individuell verfügbarer Methoden illustriert, das seinerseits dynamisch aus der Methodenliste generiert wird. Dadurch wird der Wartungsaufwand bei Erweiterungen oder Änderungen, z.B. im Zusammenhang mit den nachfolgend beschriebenen Konstrukten der Klassen und Vererbung erheblich reduziert.

Die Kombination mit definierten Ereignissen in Form von attached-procedures (Triggerfunktionen) erlaubt funktional datengetriebene und damit wartungsfreundliche Anwendungen. Dadurch können Methoden nicht nur direkt, wie im oben beschriebenen, einfachsten Fall des Popup-Menüs, sondern auch indirekt aktiviert werden, z.B. in Abhängigkeit von Wertänderungen einzelner Slotbelegungen. So lassen sich etwa Grenzwertvergleiche im Rahmen eines Exception-Reporting auslösen, die im Falle von Verletzungen "von sich aus" visuelle Hinweise incl. ergänzender Abrufmöglichkeiten für Details erzeugen.

Eine Analyse der solchermaßen integrierten Informationsfacetten ergibt, daß diese sowohl ihrerseits eine komplexe Objektstruktur aufweisen als auch, daß einzelne Bestandteile für mehrere Objekte in Betracht kommen. So werden Verantwortlichkeiten mit Name, Abteilung, Telefon usw. je Person für mehrere Objekte relevant sein, Margengruppen aus alternativen Ober- und Untergrenzwertpaaren partiell global Wirkung besitzen. Aus Gründen der Wartungseffizienz aber auch zur Abbildung organisatorischer Zuständigkeiten, bspw. in Bezug auf die Festlegung der Grenzwerte, ist es daher notwendig, zusammengesetzte, konkrete Objekte beliebiger Tiefe bilden zu können. Die gewünschte Zusammenführung prinzipiell eigenständiger Objekt(attribut)e kann im Rahmen des objektorientierten Paradigma konstruktmäßig unterschiedlich realisiert werden:

- ❒ Durch die Spezifikation von Unter-Objekten mit der Relation "ist-Teil-von", wie bspw. bereits in der Expertensystem-Shell NexpertObject gesehen (vgl. Kapitel 6.3.2.1),
- ❒ durch Slots vom Typ "Objekt", wie bspw. in der Expertensystem-Shell KappaPC (vgl. Kapitel 6.3.2.1) oder
- ❒ durch multiple Vererbung (s. Kapitel 8.1.2.3).

Voraussetzung für den Nutzen ist dabei in allen Fällen, daß die geschaffene Objektverknüpfung funktional einen transparenten Zugang zu den Methoden und damit den Dateninhalten der "verbundenen" Objektkomponenten bewirkt. D.h., die Implementierung der Relation "ist-Teil-von" muß im Falle hierarchischer Objektstrukturen quasi Anfragen an ein Master-

Objekt, die nicht von dessen Schnittstelle abgedeckt werden, an alle Subobjekte "weiterreichen"; das Objektnetzwerk muß praktisch über eine konsolidierte, virtuelle Schnittstelle verfügen. Diese Funktionalität ist leider in den wenigsten objektorientierten Sprachen oder Entwicklungstools vorhanden.

Eine Anwendungsentwicklung ausschließlich auf Basis dieses Objekt-Konstrukts bleibt aber in allen bisher dargestellten Fällen immer auf konkrete Ausprägungen mit individuellen Wertbelegungen beschränkt. Mehrfachverwendung setzt hier neben identischen Strukturen auch identische Werte voraus. Diesem Manko begegnen die Konstrukte der Klassenbildung und Vererbung.

8.1.2.2. Klassen

Durch die Einführung des Klassenkonstrukts wird das Leistungsspektrum abstrakter Datentypen adaptiert, indem jedes konkrete Objekt sich als "Abkömmling" oder Vertreter eines Prototyps darstellt. Gemeinsame Strukturen und Funktionalitäten ähnlicher Objekte müssen nur einmal als Klasse definiert werden und werden durch Instanziierung auf konkrete Objekte "übertragen" (vererbt). Das gleiche Prinzip der Vererbung (vgl. Kapitel 8.1.2.3) findet auch zwischen Klassen statt, deren konzeptionelle Musterbeschreibung im Interesse der Redundanz-Minimierung als Klassenhierarchie schrittweiser Spezialisierungen abgebildet wird. Dabei werden Attribute und Funktionen entsprechend ihrem Gültigkeitsbereich auf jeweils höchst möglicher Ebene angesiedelt.

Im Beispiel aus Abb. 8-2 werden Slotstruktur (ohne oder mit prototypischen Werten) und Methoden etwa als Klasse "Information" definiert, von der sich dann neben der konkreten Ausprägung "DschnUmsatzÄnd_1_91" beliebig viele weitere Objekt-Instanzen, z.B. für andere Quartale oder betriebswirtschaftliche Größen generieren lassen. Speziellere Objekte, z.B. regionaler Art, knüpfen an entsprechende Subklassen an, die alle Eigenschaften der Oberklassen incl. evtl. Modifikationen (Redefinitionen) vereinen.

Gleichzeitig implizieren die Klassenhierarchie sowie Klassen-Instanzen-Beziehung für (Führungs-)Informationssysteme die effiziente (dynamische oder datengetriebene) Errichtung einer Informations-Selektion mit integrierter Drill-Down-Funktionalität. Abb. 8-3 zeigt exemplarisch eine derartige Implementierung. Die Klassen sind dabei auf jeder Ebene mit "Meta-"Informationen über Unter- bzw. Oberklassen und zugehörige konkrete Objekte versehen worden, die funktional als (multiple) Selektionsmenüs abgerufen werden können und zur Anzeige tiefer- bzw. höherliegender Klassen oder Objekte führen. Dadurch wird eine beliebige, vom Benutzer bestimmbare Navigation durch das Informationsnetz realisiert.

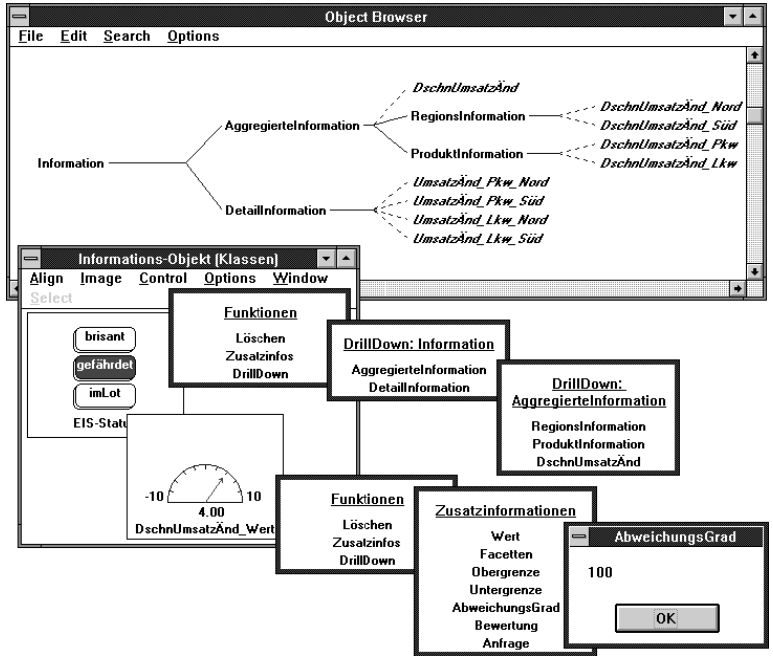


Abb. 8-3: Einsatzbeispiel des Klassen-Konstrukts bei EIS

Durch Kopplung des (manuellen) Abrufs mit z.B. Ereignissen der Grenzwertverletzung lassen sich so leicht auch strukturell datengetriebene Informationssysteme realisieren, wie sie im Zusammenhang mit Executive Information Systems (EIS) mit dem Charakter der Frühwarnung benötigt werden.

Im Beispiel der Abb. 8-3 läßt sich für das aggregierte Objekt EIS-Status neben den nun als Menügruppe Zusatzinfos zusammengefaßten objekt-unmittelbaren Methoden ein zweites, aus der Klassenhierarchie dynamisch generiertes Untermenü "Drill-Down" abrufen. Dieses führt über weitere analog generierte Menüs durch den Klassen-/Instanzenbaum, wobei Klassen jeweils über ihre vor- bzw. nachgelagerte Struktur Auskunft geben, Instanzen zur Öffnung eines weiteren Fensters führen. Im Beispiel wurde die "Dschn.UmsatzÄnderung" gewählt. Für jedes so "angesprungene" Objekt gilt natürlich die gleiche "ererbte" bzw. individuell generierte Funktionalität, so daß bspw., wie dargestellt, beliebige Zusatzinformationen abgerufen werden können.

Die Art der "Übertragung" von Struktur- und Verhaltenseigenschaften zwischen Klassen und Unterklassen sowie auf Objektinstanzen hat erhebliche Auswirkungen auf die Entwicklungs- und Wartungseffizienz. Während für eine effiziente Erstentwicklung die "kopierende" Wiederverwendung der Strukturen und Funktionen prototypischer Klassen (vgl. [Lieberman 86, Prototypical Objects]) vielleicht günstigstenfalls noch ausreichend sein mag, so erfordert die effiziente Wartung, d.h. Änderung und Weiterentwicklung weiter gehende Eigenschaften, die zum Konstrukt der Vererbung überleiten.

8.1.2.3. Vererbung

Vererbung als Konstrukt zur Übertragung von Struktur- und Verhaltens-Eigenschaften zwischen Klassen und konkreten Objektinstanzen verfolgt je nach Ursprungsdomäne zwei unterschiedliche Primärziele:

- ❒ Hauptaspekt der Domäne objektorientierter Programmierung ist die Realisierung von Mehrfach- oder Wiederverwendung von Softwareteilen,
- ❒ Hauptziel der Wissensrepräsentation in der Künstlichen Intelligenz ist die konzeptionelle Strukturierung von Information durch Generalisierung und Spezialisierung.

Beide Aspekte sind für Führungsinformationssysteme relevant und nützlich. Dabei ist Vererbung ähnlich komplex wie Nichtprozeduralität, so daß Differenzierungen nach verschiedenen Kriterien zu einer vollständigen Beschreibung notwendig sind (vgl. [Nierstranz 89, Object-Oriented Concepts], S. 6).

- ❒ *Charakter von Erbe und Erblasser*
  Unterschiede der nachfolgenden Kriterien korrelieren häufig damit, ob Vererbung zwischen Klassen im Sinne der hierarchischen Spezialisierung der Konzeptwelt, zwischen Klassen als Prototypen und konkreten Objektinstanzen oder zwischen (konkreten) Prototypinstanzen und daraus abgeleiteten Instanzen stattfindet. Dieser letzte, nicht zwangsweise hierarchische Fall wird als "Vererbung" durch Prototyping und Delegation bezeichnet (vgl. [Witt 92, Objektorientierte Programmierung], S. 128).

  Für die Abbildung von Informationsstrukturen ist zudem eine zu den Vererbungsmechanismen zwischen Klassen analoge Vererbung zwischen Objektinstanzen sinnvoll. Auf diese Weise können dann Attribute zwischen zusammengesetzten Objekten transferiert werden. Bspw. könnte so das konkrete Objekt "DschnUmsatzÄnd." als weiteren Bestandteil eine von mehreren Instanzen für Margen einer Klasse Grenzwerte erhalten. Die speziellen Wertepaare würden dann von der Instanz geerbt, die zugehörige Funktionalität des Exception-Reporting von deren Klasse Grenzwerte.

- ❒ *Gegenstand der Vererbung*
  Hierbei ist zu unterscheiden, ob strukturelle (Slots bzw. Klassen-/Instanzvariablen), inhaltliche (prototypische bzw. "globale" Slotwerte) oder funktionale (Methoden) Objektattribute weitergegeben werden.
- ❒ *Funktionalität von Vererbung*
  Dies bezeichnet die Art und Weise der Weitergabe von Attributen, d.h. ob eine (eigenständige) Kopie wie im Falle der Erzeugung von Instanzvariablen bei der Instanziierung konkreter Objekte entsteht, oder lediglich die (lesende) Nutzungsmöglichkeit (parametrisierter) Methoden des übergeordneten Objekts möglich ist.
- ❒ *Dynamik der Vererbung*
  Dieser Aspekt bezeichnet den Zeitpunkt, zu dem die Weitergabe stattfindet bzw. stattfinden kann. Im statischen Fall ist sie auf den Erstellungszeitpunkt von Objektinstanzen beschränkt, im dynamischen Fall können Struktur-, Wert- und Funktionsänderungen auch während der Laufzeit von "Erben" registriert und berücksichtigt werden.
- ❒ *Kardinalität*
  Hier wird zwischen einfacher und multipler Vererbung unterschieden, d.h. ob eine Klasse oder ein konkretes Objekt jeweils nur einen oder mehrere Vorgänger haben kann. Dabei auftretende Konflikte müssen durch geeignete Vorkehrungen geregelt werden, z.B. Linearisierung, Umbenennung, Qualifizierung mit dem Klassennamen, Durchschnittsbildung, Überdeckung usw. (vgl. [Witt 92, Objektorientierte Programmierung], S. 121 ff.).

  Für die Gestaltung von (Führungs-)Informationssystemen ergeben sich zahlreiche, sinnvolle Anwendungsfelder für multiple Vererbung. Bspw. kann die Funktionalität der bereits häufig bemühten, aggregierten Kennzahl "durchschnittliche Umsatzänderungen je Quartal" dadurch realisiert werden, daß sie als von drei Kennzahl-Klassen "abstammend" definiert wird (vgl. Abb. 8-4): Durchschnitts-Kennzahlen, Abweichungs-Kennzahlen und Zeitreihen-Kennzahlen (vgl. [Rieger 90, EIS-State of the art], S. 355-357).

  Jede Kennzahl-Klasse kann dann über individuelle Sätze geeigneter Methoden für Präsentationsgraphiken verfügen, die jeweils in einem gleichen Klassenattribut "registriert" sind, das alle von einer gemeinsamen Oberklasse "Kennzahlen" strukturell erben. Eine ebenfalls geerbte Methode "Graphiken" kann dann in Verbindung mit multipler Vererbung die Vereinigungsmenge zur Laufzeit bilden und in Form eines Auswahlmenüs dem Benutzer anbieten. Je nach Auswahl wird dann die passende Methode der jeweiligen Unterklasse erbend aktiviert und an das konkrete Objekt gesandt.

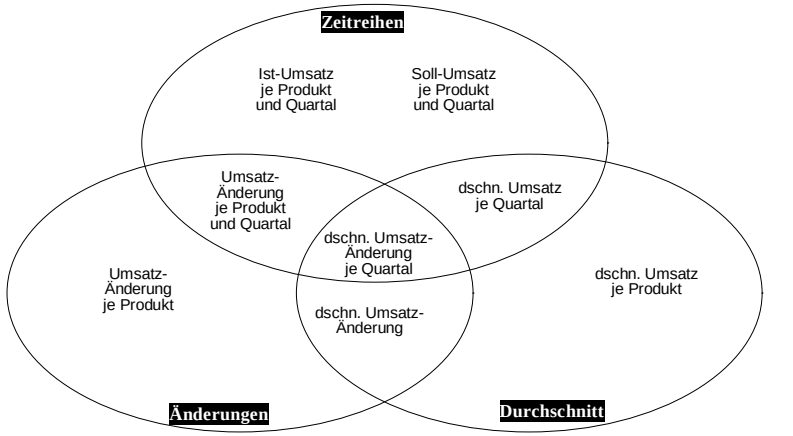


Abb. 8-4: Beispiel generischer Informationsobjekte durch multiple Vererbung

8.1.2.4. Message Passing/Polymorphismus/Dynamisches Binden

Elementarer, die vorher beschriebenen Konstrukte erst wirksam werden lassender Bestandteil objektorientierter Programmierung ist die (ausschließliche) Aktivierung von Methoden durch Nachrichtenaustausch (message passing) zwischen Objekten. Im Gegensatz zum Prozeduraufruf prozeduraler Sprachen verfügen die aktivierten Methoden nicht nur über ausschließlich lokale Modifikationskompetenz, sondern auch über alleinige Implementierungskompetenz. Diese Aufspaltung oder Verteilung konventioneller Prozedur-Verantwortlichkeiten ermöglicht das sogenannte Überladen (overloading) von Nachrichten, eine Form des Polymorphismus: gleich(namig)e Nachrichten können objektspezifisch unterschiedlich ausgeführt werden.

Der resultierende, generelle Vorteil erhöhter Wartungseffizienz kommt bei naturgemäß dynamischen Führungsinformations- bzw. -unterstützungssystemen aufgrund der stark heterogenen, kompositionellen Struktur der Informationsobjekte besonders zum Tragen. Typisches Beispiel ist etwa die für die überproportional vertretenen Aspekte von Dialog, Navigation und Kommunikation häufig benötigte Nachricht "Zeige_Information", die in der objekt- bzw. klassen-verantwortlichen Realisierung in Auswahllisten, (Kreuz-)Tabellen, Grafiken, Bild- oder Sprachausgaben usw. resultieren kann.

Die beschriebenen Konzepte des Überladens und der Vererbung werden allerdings erst in Verbindung mit dynamischem Binden, d.h. der Auswahl des entsprechenden Maschinencodes zur Lauf- bzw. Ausführungszeit realisierbar. Nachteile ergeben sich in Bezug auf Laufzeitverhalten und eingeschränkte Testmöglichkeiten.

8.2. Daten- und Informations-Verzeichnisse (Repositories)

8.2.1. Das Konzept

Ein Beitrag zur Lösung der Problembereiche Entwicklungseffizienz, unternehmensweite Standardisierung sowie Anpassung und Wartung über den Lebenszyklus von RAP-Anwendungen kann vom Konzept des Repository bzw. der Entwicklungsdatenbank ausgehen (vgl. [Habermann 93, Repository], S. 15 ff.). Als zentraler, integrativer Bestandteil bspw. der computergestützten Softwareentwicklung (CASE) sammelt ein Repository die Spezifikationen, dokumentierenden Beschreibungen und Zusammenhänge aller Elemente, die im Laufe der Phasen einer Anwendungsentwicklung erstellt, weitergegeben bzw. benutzt werden. Es kann somit als gemeinsame Archivierungs- und Kommunikationsinstanz aller am Softwareentwicklungsprozeß beteiligten Entwickler sowie der von ihnen eingesetzten Entwicklungswerkzeuge interpretiert werden.

Das Repository-Konzept vereinigt somit die Aspekte von 3 ursprünglich getrennt unterstützten Entwicklungskonzepten (vgl. [Stülpnagel 91, Repositories], S. 11):

- *Datenmanagement-Aspekt*
  Zum Zwecke der redundanzfreien Verwaltung von Datendefinitionen und -strukturen wurden bereits in den 70er Jahren sog. Data-Dictionaries entwickelt, die neben der zentralen Speicherung und unmittelbar damit zusammenhängender Verwaltungsfunktionen zusätzliche Informations- und Weiterverarbeitungsdienste leisteten. Dazu gehören insbesondere Online-Auskunfts- und Abfragefunktionen, die Generierung von Schema-Definitionen in den Datendefinitionssprachen (DDL) verschiedenster Datenbanksysteme oder Programmiersprachen, z.B. der DATA SECTION von COBOL-Programmen. Zusammen mit Analyse- oder Reengineering-Funktionen, mit denen Data-Dictionary-Einträge umgekehrt aus vorhandenen Programmen oder DDL-Spezifikationen erzeugt werden können, entwickelten sie sich so von anfänglich passiven hin zu aktiven Data Dictionaries.
- *Softwareengineering-Aspekt*
  Trotz zahlreicher Hilfestellungen im Rahmen der Softwareentwicklung genügen die Darstellungsmöglichkeiten und vor allem Dienstefunktionen klassischer Data Dictiona-

ries nicht den Anforderungen einer durchgängigen, ganzheitlichen, rechnergestützten Dokumentation sowohl des Ergebnisses als auch des Prozesses der Anwendungserstellung. So kann bspw. der Verwendungsnachweis eines klassischen Data Dictionary in Verbindung mit erweiterten Entity-Klassen wie Anwendung, Bildschirmmaske oder Programm zwar den erreichten Stand der Spezifikation des zu entwickelnden Systems repräsentieren, es fehlen jedoch so entscheidende Dinge wie Entstehungsgeschichte, Interaktionslogik usw., kurz das gesamte Modell der Problemlösung. Damit entfallen wichtige Unterstützungsbereiche wie automatisierte Überprüfung auf Konsistenz, Vollständigkeit und Redundanzfreiheit.

❒ *Dokumentationsaspekt*
Um ein Repository aktiv zur anpassenden oder ergänzenden Weiterentwicklung darin gespeicherter Anwendungssysteme resp. deren Komponenten einsetzen zu können, sind neben den strukturierten Elementen auch deskriptive Informationen über Projektdaten, wie z.B. Ziele, Nutzerkreise, Entwicklungsversionen usw. abzulegen, um diese über geeignete Dokumentationswerkzeuge pflegen bzw. Bibliotheksfunktionen abrufen zu können. Da die zu dokumentierenden Inhalte stark heterogenen Charakter aufweisen, d.h. sich vom Fließtext z.B. eines Pflichtenhefts bis zu graphischen Ablauf- oder Interaktionsmodellen erstrecken, die spezialisierte Werkzeuge erfordern, bietet sich als Lösung an, dem Repository lediglich die Rolle eines Referenzsystems zuzuweisen und die Speicherung der eigentlichen Dokumente den jeweiligen eingesetzten Werkzeugen in deren speziellem und damit effektiveren Format zu überlassen.

Zusammenfassend werden die Anforderungen an ein solches Metadatenbanksystem wiefolgt beschrieben (vgl. [Eicker 91, Metadatenbanksystem]):

❒ *Speicherung von Metadaten*
❒ *Versionsverwaltung und Statusverwaltung*
❒ *Integrität der gespeicherten Metadaten*
❒ *Anwendungssteuerung*
❒ *Zugriffsverwaltung*
❒ *Schnittstellen*

8.2.2. Informationsmodelle für Repositories

Welche Objekte, bzw. genauer Objektstrukturen und -beziehungen, in einem Repository gespeichert werden können, wird in Analogie zur Vorgehensweise in Datenbanksystemen auf abstrakterer Ebene durch ein Metamodell, häufig (Repository-)Informationsmodell genannt, festgelegt. Es bestimmt im Detail, welche Objekttypen mit welchen Attributen, also Objektstrukturen dokumentiert werden können, sowie welche Beziehungstypen oder Verknüpfungen zwischen diesen, also welche Objektbeziehungen darauf definiert werden können.

Unabhängig von der späteren physischen Speicherung wird zur Modellierung dieses Metamodells zumeist die Entity/Relationship-Methode (ERM) nach Chen (vgl. [Chen 76, Entity-Relationship Model]) verwendet, deren Basiskonstrukte aus Gründen der besseren Beschreibbarkeit schnell erweitert wurden. So sind in diesen erweiterten Formen auch Relationships zwischen Entities und Relationships oder zwischen Relationships zugelassen, genauso wie Attribute für Relationships selbst. Zwar können damit die Darstellungsvielfalt und Aussagekraft der Informationsmodelle erheblich gesteigert werden, insbesondere was die Integration der verschiedenen Modellsichten der Anwendungsentwicklung (Datenmodell, Funktionsmodell, Organisationsmodell, Steuerungsmodell, Vorgehensmodell) angeht, gleichzeitig vergrößert sich aber auch deren Komplexität beträchtlich - bis in die Nähe eines Isomorphismus, d.h. der Identität zum realen Modell. Hinzu kommen Schwächen bezüglich der Erweiterungsfähigkeit und Wiederverwendbarkeit. Dies wurde durch zusätzliche Erweiterungen in Form spezieller Beziehungstypen zur Abbildung der aus der Datenmodellierung bekannten Konstruktionsoperatoren auszugleichen versucht: (vgl. [Scheer 90, Wirtschaftsinformatik], S. 33ff.)

❒ *Klassifizierung* zur Identifikation von Objekten gleicher Attributstruktur zu Objekttypen
❒ *Generalisierung/Spezialisierung* zur Bildung von über- bzw. untergeordneten Objekttypen
❒ *Aggregation* zur Zusammenfassung unterschiedlicher Objekttypen zu einem neuen Objekttyp im Sinne einer "besteht-aus"-Relation und
❒ *Gruppierung* als listenbildender Objekttyp einer Menge gleichartiger Objekttyp-Ausprägungen.

Diese Erweiterungen stellen jedoch nichts anderes dar, als eine implizite Übernahme originär objektorientierter Prinzipien, ohne jedoch deren gesamtes Leistungsspektrum ausnutzen zu wollen.

Ein konsequenterer Ausweg im Sinne der Komplexitätsreduktion und damit eine Alternative zu diesen schrittweisen Erweiterungen der Beschreibungsmethodik für das Informationsmodell kann im Wechsel der Beschreibungsmethodik selbst und damit in objektorientierten Repositories gesehen werden. Ein Beispiel hierfür wird in [Ljubojevic 91, objektorientiertes Informationsmodell] auf Basis einer produktmäßigen Implementierung, dem CDD/Repository von DEC (vgl. [Habermann 93, Repository], S. 151 ff.), gegeben (vgl. Abb. 8-5):

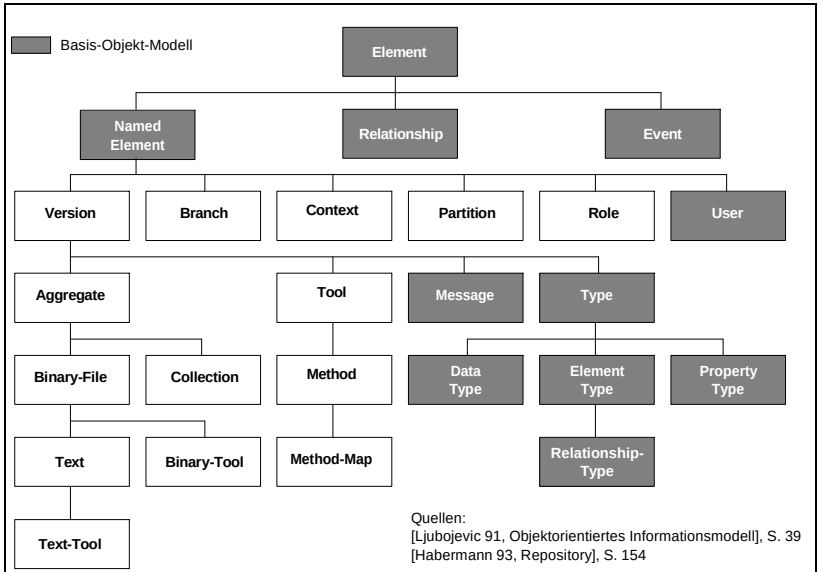


Abb. 8-5: Objektorientiertes Repository-Klassenhierarchie-Modell (Auszug)

Das Informationsmodell repräsentiert sich dabei als Klassenhierarchie, die aus der Integration folgender Teilmodelle entstanden ist, die ihrerseits entsprechend der verschiedenen Sichten der am Softwareentwicklungsprozess beteiligten Entwicklerklassen bzw. Entwicklungsaufgaben gebildet wurden. Im einzelnen sind dies (vgl. auch [DEC 90, ATIS]):

- ❒ *Basis-Objektmodell*
- ❒ *Versionsmodell*
- ❒ *Konfigurationsmanagementmodell*
- ❒ *Arbeitsflußmodell*
- ❒ *Werkzeugintegrationsmodell*
- ❒ *Sicherheits- und Zugangskontrollmodell*
- ❒ *Namensvergabemodell und*
- ❒ *Transaktionsmodell.*

Die letzten 3 Modelle dienen dabei ausschließlich dazu, die ersten 5 Teilmodelle in einer verteilten Mehrbenutzerumgebung umsetzen und anwenden zu können. Die Teilmodelle 2-8 stellen lediglich Teilmengen des Basis-Objektmodells dar.

Ein weiter reichendes Beispiel, das die Bedeutung und den Nutzen der aktiven Integration

von Repositories in und für Anwendungssysteme verdeutlicht, ist das R/3-System von SAP (vgl. [Habermann 93, Repository], S. 165 ff.). Da zum R/3-System mittlerweile auch eine EIS-Komponente (SAP/EIS) gehört, ist zumindest die Basis für nachfolgend beschriebene Nutzungsmöglichkeiten des Repository-Konstrukts durch Führungsinformationssysteme gelegt.

Als Nutzungsmöglichkeiten für Führungsinformations- und -unterstützungssysteme lassen sich bspw. vorstellen:

- ❑ der direkte Zugriff auf definitorische Datenbeschreibungen des Data-Dictionary zur Erklärung von Kennzahlen aus dem EIS heraus,
- ❑ die direkte Verwendung des Informationsmodells bezüglich Struktur und Inhalt bei der Formulierung von adhoc-Anfragen im EIS,
- ❑ die Auslagerung der Verwaltung und Überwachung von Zugriffsberechtigungen

u.v.a.m..

8.3. Verteilte Verarbeitung

8.3.1. Relevante Prinzipien und Konstrukte

Die Zerlegung und Verteilung komplexer betrieblicher Aufgaben und Arbeitsabläufe auf verschiedene Aufgabenträger unterschiedlicher Hierarchiestufen ist zentrales Prinzip der Unternehmensführung und -organisation und findet ihren Ausdruck in Aufbau- und Ablauforganisation. Es findet sich unter den Schlagworten Dezentralisierung, verteilte Verarbeitung (distributed processing) und Downsizing im Bereich rechnerunterstützter Abläufe. Bei den ursprünglichen, methodisch konventionellen Anwendungen standen primär Gründe der optimalen Lastverteilung, kostengünstigen Nutzung von Spezialperipherie oder -software im Vordergrund. Dementsprechend zeichnen sich die unterschiedenen Formen horizontaler und mehr noch vertikaler Verarbeitung dadurch aus, daß die Aufgabenverteilung a priori exakt festgelegt ist und zumeist rigide zentral gesteuert und koordiniert wird (vgl. [Stahlknecht 91, Wirtschaftsinformatik], S. 153 ff.). Kramer zählt als Vorteile verteilter Verarbeitung auf: reduzierte Kosten, Zwang zu modularer, weniger komplexer Software, dadurch erhöhte Flexibilität und Erweiterbarkeit, höhere Robustheit (Ausfallsicherheit), verbesserte Leistung (Antwortzeiten) und lokale Anpaßbarkeit (vgl. [Kramer 89, verteilte Systeme]).

Mit der Aufnahme des Verteilungsprinzips durch die Künstliche Intelligenz Forschung Anfang der 80er Jahre hat eine tiefer gehende Interpretation Einzug gehalten (vgl. [Corkill 87,

Merkmale	Konventionelle verteilte Systeme	DAI-Systeme
Aufgaben	wohldefiniert	nicht im Detail bekannt
Probleme	a priori partitioniert	Partitionierung Bestandteil der Lösung
Koordinierung	extern; per Design	intern; per Kooperation
Kontrolle & Kommunikation	rigide	flexibel
Kooperation	implizit; passiv	explizit; aktiv
System-Modell	Knoten im Netz	Akteur (Agent) in Gesellschaft
Fragestellungen	Kommunikation; Synchronisation	Akteur (Agent)-Modellierung; Verhandlungen

Abb. 8-6: Gegenüberstellung konventioneller und wissensbasierter (DAI) verteilter Systeme

distributed problem solving]; [Durfee 89, Distributed Problem Solving]; [Martial 92, Verteilte Künstliche Intelligenz]). Abb. 8-6 stellt wesentliche Unterscheidungsmerkmale dieser gewandelten Sichtweise der verteilten künstlichen Intelligenz (distributed artificial intelligence, DAI) zusammenfassend dar (vgl. [Sundermeyer 92, verteilte WBS]).

Wesentlich dabei sind, daß die Problemzerlegung und -verteilung auf kompetente, häufig untereinander konkurrierende, spezialisierte Aufgabenlöser, hier selbständige Akteure oder Agenten genannt, Bestandteil des Lösungsprozesses sein kann, dieser kooperativ abläuft und die Koordination flexibel und dezentralisiert vonstatten geht. Die Relevanz der in der KI-Welt in der Folgezeit entwickelten theoretischen Problemlösungskonzepte und softwaretechnischen Kooperationskonstrukte für Führungsinformations- und -unterstützungssysteme resultiert aus dem empirisch belegten, hohen Kommunikations-, Kooperations- und Koordinationsbedarf bei Führungsentscheidungen (vgl. Kap. 2.2).

Elementare Konzepte der DAI sind Agenten, Kooperation und Koordination:

- Ein *Agent* ist eine - in den Termini der DAI intelligente, d.h. Wissen verarbeitende - Problemlöseeinheit, die autonom und zielorientiert vorgeht, Wissen über den Problemkontext und die eigene Lösungskompetenz besitzt und kooperationsfähig ist, d.h. aktiv die Kommunikation mit anderen Agenten sucht (vgl. [Albayrak 92, Kooperatives Problemlösen], S. 52). Der Agentenbegriff hat sich aus den Aktoren (actors) der objektorientierten Welt entwickelt, bei denen Parallelität der Verarbeitung und Kommunikation durch Nachrichtenaustausch (message passing) im Vordergrund standen (vgl. [Hewitt 77, Control Structures]; [Agha 86, Actors]).

- Unter *Kooperation* wird das koordinierte Handeln von Agenten verstanden, um ihre individuellen oder von außen vorgegebenen Ziele zu erreichen (vgl. [Sundermeyer 92, verteilte WBS]). Dabei wird zwischen Arbeits- und Ergebnisteilung (task- vs. result-sharing) unterschieden. Während task-sharing die Delegation von mangels Kapazitäten oder Kompetenz nicht bearbeitbaren Teilaufgaben an andere Agenten bezeichnet, entspricht result-sharing einer Art datengetriebener Koordinierung, bei der die Bearbeitung hierarchisch abhängiger Teilaufgaben erst durch Ergebnisse anderer Teilaufgaben initiiert wird.
- *Koordination* umfaßt alle zur Kooperation eingesetzten Mechanismen der (Teil-)Aufgabenbildung und -verteilung auf die Agenten zu Beginn und während des Lösungsprozesses, Konfliktlösung und Inkonsistenzbeseitigung, z.B. durch Verhandlungen (vgl. [Steiner 92, Kooperation]).

8.3.2. Klassifizierung

Nach Bond und Gasser (vgl. [Bond 88, DAI]) werden in der DAI zwei Teilgebiete unterschieden,

- das Verteilte Problemlösen (Distributed Problem Solving, DPS) und
- die Multi-Agenten-Systeme (Multi-Agent-Systems, MAS).

Multi-Agenten-Systeme sind eher auf die verteilte Bearbeitung einer Problempalette durch über den Lösungsprozeß hinaus existente, primär homogene und untereinander konkurrierende Agenten ausgerichtet, die sich selbst organisieren bzw. koordinieren. Verteiltes Problemlösen stellt im Gegensatz dazu die verteilte Lösung eines bestimmten Problems in den Mittelpunkt. Die Zerlegung in Teilaufgaben und deren Verteilung sind Bestandteil des DPS und werden i.d.R. durch einen speziellen Koordinierungs-Agenten übernommen. Andere Agenten sind im Unterschied zu MAS nur über den Koordinator (indirekt) erreichbar.

8.3.3. Ausgewählte Konstrukte des kooperativen Problemlösens

8.3.3.1. Contract-Net

Das auf dem Konzept der Vertragsverhandlungen basierende Contract-Net-Modell der Kooperation zwischen Agenten ist der Gruppe des task-sharing zuzuordnen. Verträge, d.h. Übereinkünfte ausgelagerter Teilaufgaben kommen durch ein normiertes, typisiertes Nachrichtenprotokoll zwischen Auftraggeber-Agent und sich bewerbenden Auftragnehmer-Agenten zustande (vgl. [Smith 80, Contract-Net-Protocol]).

Für eine Übertragung auf Führungsinformationssysteme ist besonders dieses Nachrichtenprotokoll höherer Ebene interessant. Wenn die Führungskraft als potentieller Auftraggeber von Informationsnachfragen oder Detailanalysen diese nicht als undifferenzierte Kommentierungen gezielt versendet, sondern durch eine multi-dimensionale Qualifizierung normierter Merkmale charakterisiert, kann die Auswahl von Adressat sowie dessen Vorprüfung bezüglich Kompetenz, Machbarkeit oder Zeitaufwand und Kosten unter Hinzuziehung der Problemdaten teilautomatisiert werden.

8.3.3.2. Blackboards

Demgegenüber kann das Blackboard-Konstrukt der Kooperationsform des result-sharing zugerechnet werden. Hierbei findet die Kommunikation zwischen den Agenten indirekt und "unpersönlich" über ein shared-memory statt. Aufgrund von Änderungen in diesem, z.B. aufgrund von eingestellten Bearbeitungsergebnissen anderer Agenten, bestimmen die Agenten selbst, wann und wie sie aktiv werden (vgl. [Engelmore 88, Blackboard system definition], S. 1 ff.).

Dieser Ansatz läßt sich auf Führungsinformationssysteme als Erweiterung des Trigger-Prinzips übertragen, indem Zustandsänderungen des Blackboard, das bspw. den aktuellen Informationsstand zu einem bestimmten Unternehmensbereich oder -problem repräsentiert, nicht per se zu Nachrichten oder Aktionen führen, sondern von potentiellen Analyse-, Bewertungs- oder Alternativengenerierungs-Agenten beobachtet werden, die dann komplexere Aktionen ausführen oder anstoßen können. Der Vorteil ist darin zu sehen, daß die Reaktionsweise des verteilten Systems nicht a priori bis ins Detail bekannt sein muß, ein für Informationsbeschaffungs-, verarbeitungs- und Entscheidungsprozesse im Führungsbereich charakteristisches Merkmal.

8.4. Hypertext/Hypermedia

8.4.1. Relevante Basiskonstrukte

Hypertext und Hypermedia haben sich mit wenigen Ausnahmen bislang weitgehend getrennt vom Anwendungsbereich der Führungsinformationssysteme entwickelt, obwohl ihre Konstrukte für die Flexibilisierung und Individualisierung des Informations-Verknüpfungsbedarfs von EIS prädestiniert erscheinen (vgl. [Nastansky 92, Hypermedia], S. 123). Conklin und Seidensticker listen dementsprechend folgende, historische Anwendungskategorien auf (vgl. [Conklin 87, Hypertext], S. 9 ff.; [Seidensticker 90, Hypermedia], S. 43 ff.):

- ❒ *Macro Literary Systems* zur Unterstützung großer Online-Bibliotheken,
- ❒ *Problem Exploration/Resolution Tools* zur Ideensammlung und -strukturierung,
- ❒ *Structured Browsing Systems* zur benutzerfreundlichen, nicht-linearen Informationssuche und
- ❒ *General Hypertext Technology*, um die Basiskonzepte anwendungsunabhängig als Entwicklungswerkzeuge zur Verfügung zu stellen. In diese Kategorie fallen auch Anwendungen aus dem weiteren Umfeld von Führungsinformationssystemen, nämlich des persönlichen Informationsmanagements (PIM).

Die Attraktivität von Hypertext/Hypermedia für EIS resultiert aus folgenden Basiskonstrukten und zugehörigen Operationen:

- ❒ *Knoten*
  Sie dienen der Speicherung und Darstellung von Informationen unterschiedlichsten Typs in Form frei manipulierbarer Bildschirmfenster. Durch Typisierung der Knoten (Klassenbildung im objektorientierten Sinne) lassen sich anwendungsspezifische Funktionalitäten (Methoden) zuordnen, z.B. EMail bei Notizen, Warn-Trigger bei Grenzwertbedingungen etc..
- ❒ *Links*
  Sie dienen der Verknüpfung von Knoten oder (Teil-)Inhalten von Knoten. Diese in der Regel gerichteten, bipolaren Verbindungen können ebenfalls anwendungsspezifisch typisiert werden, über Attribut-Werte-Paare Zusatzinformationen aufnehmen, z.B. als Basis für das Informations-Retrieval, sowie durch zugeordnete Funktionen spezielle Aktionen auslösen, die bspw. über das Anzeigen des Zielknotens hinausgehen.

Typisch für Hypertext-/Hypermedia-Anwendungssysteme ist dabei, daß diese Konstrukte und Funktionalitäten nicht nur dem Anwendungsentwickler sondern insbesondere und primär auch dem Anwender zur Ausführungszeit zur Verfügung stehen. Dadurch können nicht nur - wie bei EIS-Systemen üblich - vorgefertigte, statische Informations-Verknüpfungsstrukturen verfolgt werden, sondern auch benutzerseitig individuell neue Verbindungen generiert werden, sei es temporär oder mit kontextbezogener Speicheroption über die genannten Link-Attribute. Dadurch lassen sich beliebig viele, alternative oder untereinander verknüpfte Navigationsnetze auf der Datenbasis aus Informationsknoten aufbauen.

8.4.2. Nutzenpotentiale für EIS

Über die Basiskonstrukte hinaus, die im wesentlichen eine nicht-lineare Informationsverar-

beitung ermöglichen (vgl. [Kuhlen 91, Hypertext], S. 28), wurden im Zusammenhang mit typischen Problemstellungen bei der praktischen Nutzung von Hypermedia Lösungsansätze in Hypermedia-Werkzeugen implementiert, die ähnliche, bisher schlecht oder nicht gelöste Schwierigkeiten beim Einsatz von Führungsinformationssystemen adressieren (vgl. Kap. 7.2):

- *Link-Generierung*
  Das (Wieder-)Finden relevanter Informationen in Kontexten unterschiedlicher Fragestellungen hängt entscheidend von der Art und dem Umfang der Informationsvernetzung ab. Während klassische EIS-Werkzeuge ausschließlich auf statische Verknüpfungen durch den EIS-Entwickler setzen, bieten Hypertextsysteme, wie z.B. Agenda von Lotus, teilautomatisierte Generierungshilfen an (vgl. [Nastansky 90, Hypermediabasiertes Informationsmanagement], S. 525). Die Referenzierung kann dabei durch ein vom Benutzer spezifizierbares, hierarchisches Kategoriesystem von Schlüsselworten gesteuert werden, die dann im Rahmen einer Analyse eingegebener Texte (Knoten), z.B. Annotationen, entsprechende Links auf- und - aus Sicht der Wartungsproblematik besonders interessant - auch wieder abbauen. In Bezug auf das Informationsretrieval ist in diesem Zusammenhang auch eine automatisierte Charakterisierung von Knoten und Links durch Attribute von Interesse, die ebenfalls aus den Informationsinhalten oder dem Eingabekontext abgeleitet werden bzw. im Sinne einer Teilautomatisierung dem Anwender vom System zur Charakterisierung vorgeschlagen werden.

- *Navigation*
  Die Möglichkeit, schrittweise von Knoten zu Knoten über eine Vielzahl alternativer Links durch das Informationsnetzwerk zu springen, birgt ähnlich wie das Drill-Down-Konstrukt in EIS-Systemen die Gefahr der Desorientierung ("lost in hyperspace") des Anwenders. Zusätzlich zum Retrace-Path-Konstrukt von EIS, einem Protokoll aller geöffneten Knoten des verfolgten Pfades in Form einer Menüliste, finden sich in Hypermediasystemen zumeist grafische Browser globaler oder lokaler Art. Sie bieten zwar den Vorteil, neben einer Visualisierung des aktuellen Standorts auch Sprungoptionen zu anderen, bisher nicht berührten Knoten zu eröffnen, ihr praktischer Einsatz scheitert jedoch zumeist an den Grenzen der Darstellbarkeit realer Komplexitätsgrade. Als Ausweg bietet sich eine Kombination verschiedenster Navigations-Paradigma an, vom linearen "Blättern" (Buch-Paradigma) über Verzeichnishierarchien (Roadmap-Paradigma) bis hin zu grafischen Umgebungsübersichten nach dem Fischaugen-Prinzip (local tracking maps) (vgl. [Nastansky 92, Hypermedia], S. 129).

- *Informationsretrieval*
  Die Mechanismen zur Unterstützung des Informationsretrieval in klassischen EIS-Werkzeugen sind überraschenderweise begrenzt. Sie beschränken sich zumeist auf Abfragen in originären Datenquellen oder Ausnahmebedingungen im Rahmen des Exception Reporting. Hypermedia-Werkzeuge bieten hier neben Schlüsselwort- und Volltextsuche in (textuellen) Informationsknoten vor allem Abfragen auf der Basis von Knoten- und Link-Attributen an. Dadurch lassen sich im Gegensatz zu der teilweise manuell verfügbaren Zusammenstellung beliebiger Informationsobjekte, etwa in Forest&Trees, bei entsprechender Charakterisierung (s.o.: Link-Generierung) praktisch beliebige, heterogene Informationsmengen je nach Problemstellung selektieren.

- *Multiple, heterogene Teilnetze*
  Solchermaßen selektierte Knoten und Links repräsentieren ihrerseits voll funktionsfähige Teilnetze der Hypermedia-Datenbasis. Im Zusammenhang mit einer optionalen Speicherung der Retrieval-Abfragen realisieren sie implizit die Möglichkeit, in den Grenzen der Abfragekombinatorik beliebig viele, alternative und dennoch gleichzeitig untereinander verbundene Informationsnetzwerke bzw. Navigationsstrukturen zu verwalten. Im Gegensatz zu klassischen EIS-Werkzeugen mit ihrer typischerweise starren, baumartigen Informationsstruktur eröffnet sich dem Anwender so ein in Bezug auf die aktuelle Fragestellung adäquaterer Einstieg in den Informationspool bei vollem Erhalt aller Navigationsmöglichkeiten.

8.4.3. Nutzenpotentiale von Hypermedia-Entwicklungsumgebungen

Neben den beschriebenen, mehr die Anwendungsflexibilität betreffenden Nutzenpotentialen existieren auch Vorteile in Bezug auf die Entwicklungs- und Wartungsproduktivität. So kann bereits die in der Literatur häufig genannte, behelfsmäßige Link-Typisierung auf höchstem Abstraktionsniveau wesentlich zu einer klaren Strukturierung von Führungsinformationssystemen beitragen (vgl. [Gloor 90, Hypermedia], S. 16):

- *Hierarchische Links* bilden die Hauptstruktur des EIS ab,
- *Referenzierende Links* repräsentieren Hinweise auf evtl. relevante Informationen anderer "Äste" der Hauptstruktur, differenziert bspw. nach unterschiedlichen Fragestellungen,
- *Annotative Links* betreffen ergänzende, zumeist kommentatorische und private Informationen.

Die Trennung in Linktypen fördert nicht nur die Orientierung des Anwenders während der Navigation, sondern auch die Übersichtlichkeit des Entwicklers bei wartungsmäßigen Erweiterungen.

Weitere Nutzenpotentiale leiten sich aus dem konsequenteren Einsatz objektorientierter Prinzipien ab und decken sich demzufolge mit den in Kap. 8.1 dargestellten Aspekten. Hervorzuheben sind insbesondere hierarchisch organisierte Basisobjekte, z.B. Home-Stacks, Card-Stacks, Cards, Fields, Buttons und Background in Hyper- bzw. SuperCard mit integrierten Vererbungsmechanismen zur Weiterreichung zugeordneter Funktionen (Handlers) (vgl. [Seidensticker 90, Hypermedia], S. 68 ff.).

Aufgrund der offenen Systemarchitektur lassen sich im Gegensatz zu dem fest vorgegebenen Objektvorrat in klassischen EIS-Werkzeugen, z.B. Forest&Trees oder LightShip, individuelle Informationsobjekte (Templates) konfigurieren. In einem weiteren Schritt lassen sich gar individuelle Entwicklungsumgebungen erstellen, z.B. für Anwendungen des persönlichen Informationsmanagements (PIM) (vgl. [Seidensticker 90, Hypermedia], S. 86 ff.; [Nastansky 90, Hypermediabasiertes Informationsmanagement], S. 529 ff.)). Allerdings ist festzustellen, daß auch hierbei zumeist das Spektrum objektorientierter Konstrukte, speziell in Bezug auf Klassenbildung und Instanziierung nicht voll angewandt wird, sondern allzu häufig mit Cut&Paste gearbeitet wird.

8.5. Neuronale Netze

8.5.1. Relevante Konstrukte

Die junge Informatikdisziplin der künstlichen neuronalen Netze stellt einen weiteren Versuch der Künstliche Intelligenz Forschung dar, sich der "Rechenleistung" des menschlichen Gehirns durch (vereinfachte) Nachbildung bzw. Simulation von dessen Aufbau und Funktionsweise in Form einfachster, dafür jedoch zahlreicher und extrem vernetzter Schaltstellen (Neuronen) anzunähern. Basiskonstrukte sind demzufolge Neuronen, die als selbständige Prozessoren in einem Multiprozessorsystem angeordnet werden. Einem Neuron wird dabei - entsprechend dem bisherigen Kenntnisstand bzw. Hypothesen über die Funktionsweise biologischer Neuronen - die in Abb. 8-7 dargestellte, prinzipielle interne Struktur zugeordnet (vgl. [Brause 91, Neuronale Netze], S. 36).

Es transformiert einen Vektor von Eingabeimpulsen, die jeweils aus der Umwelt oder aus anderen Neuronen aufgenommen werden, in prinzipiell drei Schritten zu einem Ausgabesignal, das wiederum an die Umwelt oder andere Neuronen weitergereicht werden kann. Die drei Schritte, deren Ausgestaltung neben der Netztopologie (s.u.) eine wesentliche Gruppe von Konfigurationsparametern für die Vielfalt neuronaler Netzwerkmodelle darstellt, haben folgende Aufgaben (vgl. [Dorffner 91, Konnektionismus], S. 16):

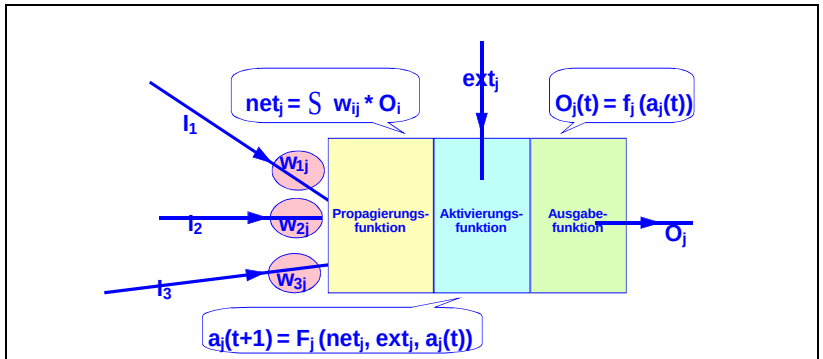


Abb. 8-7: Basisstruktur eines Neurons in künstlichen neuronalen Netzen

❒ *Propagierungsfunktion*
Sie dient der Aggregation der Eingangsimpulse zu einem Signalwert. Gebräuchliche Funktionen sind bspw. gewichtete Summe, Durchschnitt oder Produktsumme sowie Maximum oder Minimum.

❒ *Aktivierungsfunktion*
Sie übernimmt die eigentliche Schaltfunktion, d.h. ob und teilweise auch wie das aggregierte Eingangssignal weitergegeben wird. Sie kann als binäre Funktion einen harten Schwellwert, häufig als Bias bezeichnet, repräsentieren oder andere Transferverläufe aufweisen (linear, sigmoid).

❒ *Ausgabefunktion*
Sie dient schließlich der Generierung des eigentlichen Ausgabesignals und wird häufig zusammen mit der Aktivierungsfunktion abgebildet und dann als Transferfunktion bezeichnet.

Der zweite Verhaltens-bestimmende Faktor ist die Netztopologie. Hierunter sind Anzahl und Anordnung von Neuronen zwischen Ein- und Ausgabeseite des neuronalen Netzes zu verstehen. Das Spektrum (vgl. Abb. 8-8) reicht von einlagigen bis mehrstufigen Netzen, die dann mehrere verborgene Schichten, d.h. Schichten ohne Umweltkontakt, aufweisen. Dabei wird weiterhin danach differenziert, ob bei der Vernetzung ausschließlich vorwärts gerichtete Verbindungen (feedforward-Netze) oder auch Rückkopplungen auftreten (vgl. [Kinnebrock 92, Neuronale Netze], S. 25).

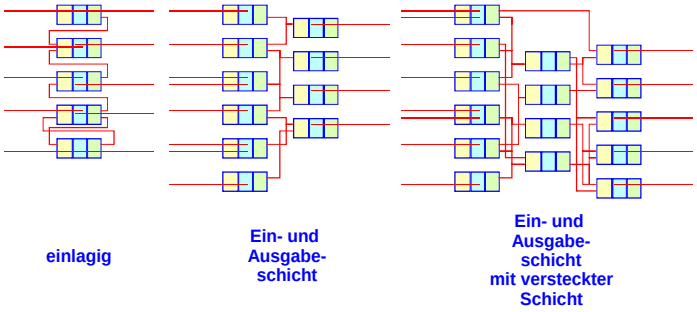


Abb. 8-8:	Beispiele neuronaler Netz-Topologien

Die letzte Gruppe von Basiskonstrukten betrifft die Mechanismen, nach denen die beiden erstgenannten Konstruktgruppen bzw. einzelne Parameter davon in der sogenannten Trainingsphase oder während des aktiven Netzeinsatzes modifiziert werden. Sie werden als Lernregeln bezeichnet und machen den adaptiven Charakter neuronaler Netzwerke aus. In der einfachsten Form beziehen sie sich auf die Veränderung der Gewichte in den Propagierungsfunktionen in Abhängigkeit von Änderungen der Ein- und (während der Trainingsphase vorzugebenden) Ausgabesignale (Hebb' sche Lernregel) bzw. Abweichungen des generierten Ausgabesignals vom gewünschten Wert (Delta- oder Widrow-Hoff-Regel). Für detailliertere Darstellungen, die den Rahmen dieser Arbeit sprengen würden, siehe bspw. [Dorffner 91, Konnektionismus], S. 248 ff.; [Kinnebrock 92, Neuronale Netze]; [Brause 91, Neuronale Netze] oder [Ritter 91, Neuronale Netze].

8.5.2. Nutzenpotentiale für EIS

Trotz intensiver, vielseitiger Forschungs- und Entwicklungsarbeiten in Bezug auf problemadäquate Netztopologien, Transformationsfunktionen und Lernalgorithmen, die insbesondere auch strukturelle Elemente der Netzarchitektur einbeziehen, muß dieser Technologie noch das Attribut des Experimentierstadiums attestiert werden. Dennoch lassen sich aufgrund erster erfolgreicher Anwendungen auch im betriebswirtschaftlichen Bereich (vgl. [Schöneburg 92, Neuronale Netze]) in begrenztem Rahmen Nutzenpotentiale beschreiben.

Für Führungsinformationssysteme von besonderem Interesse ist dabei das Anwendungsfeld der multi-kriteriellen Klassifizierung komplexer Situationen. Hierdurch können sich Alternativen zum sehr mechanistischen Exception-Reporting von EIS ergeben, das bislang praktisch auf die isolierte Bewertung vieler Einzelaspekte konzentriert ist.

Ein zweiter Ansatzpunkt ergibt sich im Bereich des Informationsselektions- und -verarbeitungsverhaltens von Führungskräften, das durch ein neuronales Netz über die Zeit akkumuliert zu situationsabhängig adaptiven Benutzeroberflächen in Bezug auf systemseitig angebotene Informationsinhalte, Darstellungs-, Analyse- und Verarbeitungsfunktionen führen kann. Diese tendenziell nachfragegesteuerte Pflege des Informationsangebots an der Oberfläche kann auch auf die Wartung der EIS-Datenbasis durchschlagen, indem bspw. häufig (zusammen) abgefragte Informationsobjekte in ihr verbleiben bzw. mit höherer Priorität versehen werden, während andere, selten oder längere Zeit nicht verwendete Daten aus dem regelmäßigen Update-Zyklus entfernt werden (vgl. [Rieger 93, EIS-Potentiale]).

9. Integrations-Konzept und -Prototyp

9.1. Architekturmodell

Ziel des nachfolgenden Darstellung ist es, ein prinzipiell multi-dimensional erweiterbares, konzeptionelles Rahmenmodell zur Entwicklung von rechnergestützten Arbeitsplätzen zu entwerfen und dessen Praktikabilität an einem Prototypsystem beispielhaft zu illustrieren. Die wesentlichen, zu flexibilisierenden Dimensionen sind:

- *Informations-Art*
  Das Spektrum rechnergestützt verarbeiteter bzw. verarbeitbarer Informationen ist zunehmend unter qualitativen Gesichtspunkten zu sehen. Dabei ist nicht nur gemeint, daß zu numerischen und textuellen Daten auch (Bewegt-)Bild, (Hand-)Schrift und Sprache hinzukommen. Im Zusammenhang mit (Führungs-)Informationssystemen kommt vielmehr heterogenen Informationsaggregaten, die aus diesen Basis-Informationsarten über mehrere Hierarchiestufen zusammengesetzt werden, wachsende Bedeutung zu. Ein bezüglich dieser Dimension erweiterbares Konzept muß die Konfigurierung individueller Informationsobjekte in Form instanziierbarer Klassen unterstützen.

- *Informations-Verarbeitungs-Methode*
  Je nach organisatorischer Ebene und Aufgabenspektrum des zu unterstützenden Arbeitsplatzes sowie Informationsart kommen unterschiedliche Verarbeitungsmethoden zur Verwaltung, Speicherung, Abfrage, Präsentation, Analyse und Bewertung in Betracht. Das Spektrum reicht von etablierten, konventionellen bis wissensbasierten Methoden und wird stetig durch infomationstechnologische Fortschritte erweitert, wie die in Kapitel 8 beschriebenen innovativen Informatik-Ansätze, z.B. neuronale Netze, belegen. Stark differierende Komplexität und Struktur der Methoden, die bis hin zur Notwendigkeit spezialisierter Hardware reichen können, z.B. Datenbankmaschine, KI-Maschine oder Transputer für neuronale Netze, legen es nahe, nicht nur spezialisierte Softwarewerkzeuge je Methode einzusetzen, sondern im Sinne einer Client-Server-Architektur auch heterogene Hardware bedarfsweise zusammenzuführen.

Abb. 9-1 stellt im Überblick die an diesen Kriterien orientierte Zielarchitektur im Überblick dar. Im Mittelpunkt der Zielarchitektur steht ein logisch zentrales Verzeichnis aller Informationsobjekte im Unternehmen. Als einsetzbare Basistechnologie kommt insbesondere das Repository-Konzept mit objektorientiertem Informationsmodell in Betracht (vgl. Kapitel 8.2). Es repräsentiert das Wissen über alle strukturellen und funktionalen Komponenten be-

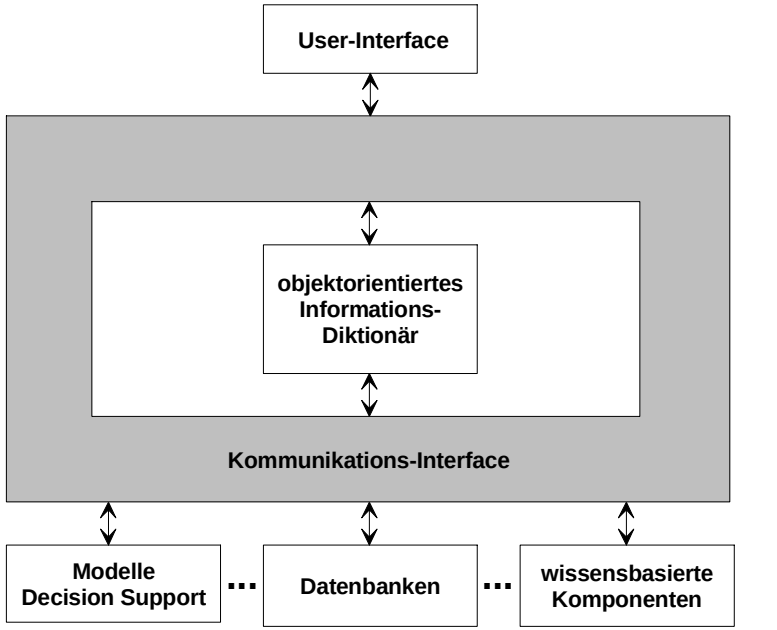


Abb. 9-1: RAP-Zielarchitektur

trieblicher IS-Anwendungen. Strukturelles Wissen bedeutet dabei, aus welchen Unterobjekten jedes Informationsobjekt besteht, und zwar rekursiv bis hinunter zu den Komponenten elementarer, d.h. nicht aus weiteren Unterobjekten bestehenden (Basis-)Informationsobjekten.

Die Elemente des Informations-Diktionärs enthalten selbst keine Daten im Sinne etwa von Produktstamm oder Auftragsdaten. Dies wird in Form von Datenquellen-Methoden und auf diese bezogenen Parametern als quasi-prozedurales Wissen hinterlegt. Die Datenquellen-Methoden stellen die Daten dann zur Laufzeit, d.h. bei Aktivierung direkt durch den Anwender oder indirekt durch ein übergeordnetes Informationsobjekt, gesteuert und getriggert durch die Informationsobjekt-spezifischen Parameter, mit Hilfe eines Kommunikations-Servers, der die Verbindung zu den methodisch geeigneten, d.h. der Informationsart entsprechenden Server-Komponenten herstellt, zur Verfügung.

Eine Server-Komponente stellt grundsätzlich Informationsobjekte beliebiger Komplexität mit deren Funktionsspektrum zur Verfügung. Das Spektrum der Informationsobjekte reicht von elementaren Größen, z.B. dem Einkaufspreis eines Zulieferteils, bis (beliebig) weit hinein in den Bereich zusammengesetzter Informationsobjekte. Dies kann sowohl ein Datensatz im herkömmlichen Sinne sein, z.B. eine Auftragsposition mit Kunden- und Artikel-Nummer, Menge und Lieferdatum, aber auch ein formatierter Bericht, z.B. eine Umsatzstatistik für ein Produkt je Region und Zeiteinheit, oder ein Analyse-, Planungs- oder Prognosemodell im klassischen DSS-Sinne. Die Komplexität wird allein durch die Art der Server-Komponente und deren Schnittstellen-Protokoll bestimmt. Je nach Grad von Komplexität und Heterogenität der Informationsobjekte können andere Server-Komponenten, gewissermaßen als Unterauftragnehmer, hinzugezogen werden.

9.2. Vorgehensmodell

Die Realisierung von Anwendungssystemen der beschriebenen Zielarchitektur muß in mehreren, hierarchisch angeordneten, in sich jeweils unabhängigen Entwicklungsschichten vollzogen werden, um den beschriebenen Flexibilitäts- und Standardisierungs-Anforderungen genügen zu können. Abb. 9-2 stellt die vorgeschlagenen Schichten in der Übersicht dar.

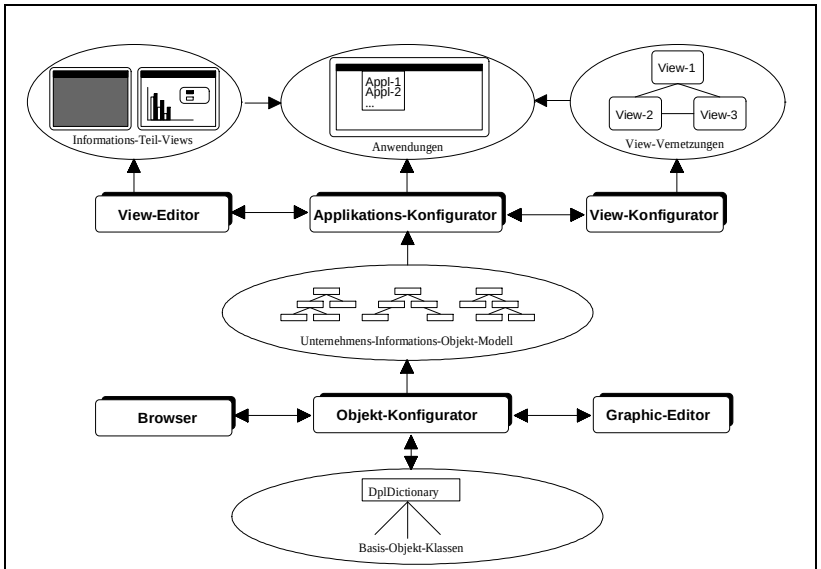


Abb. 9-2: RAP-Entwicklungsebenen

Die unterste Ebene wird durch einen hierarchischen Baum von Basis-Objekt-Klassen gebildet. Aus diesen Basis-Objekt-Klassen, gewissermaßen den elementaren (atomaren) Bausteinen von RAP-Anwendungen wird mit Hilfe eines Objekt-Konfigurators das unternehmensspezifische Informations-Objekt-Modell erstellt. Dies geschieht durch Kombination instanzierter Basisklassen. Ergebnis ist eine Menge komplexerer Objektinstanzen. Funktionale Spezifikationen bzw. Erweiterungen, die über das Angebot der Basisklassen hinausgehen, sind durch Subklassenbildung auf der Ebene der Basis-Objektklassen vorzunehmen. Hierzu wird prinzipiell dasselbe Werkzeug, nämlich der Objektkonfigurator, eingesetzt. Lediglich Zugriffsberechtigungen, Anwenderkreis und Zielbibliothek sind unterschiedlich.

Das generierte Informationsobjekt-Modell des Unternehmens kann seinerseits eine modulare Struktur aufweisen. Dazu werden Teilmodell-Hierarchien aufgebaut, die insbesondere auch verteilt gespeichert werden können. Es stellt seinerseits den Ausgangspunkt für die nächste Schicht der Anwendungsentwicklung dar.

Mit Hilfe eines weiteren Entwicklungswerkzeugs, dem Applikations-Generator werden für einzelne Informationsobjekte sogenannte Anwendungssichten (Views) generiert, die sowohl den datenmäßigen Ausschnitt als auch die graphische Repräsentation auf Seiten der Benutzerschnittstelle festlegen. Ein zweiter Schritt der Applikationsgenerierung schließlich spezifiziert das endgültige Erscheinungsbild der Anwendung durch Kombination dieser Views. Hierbei werden insbesondere die Gesamtheit abrufbarer Views sowie deren Vernetzung, d.h. die Sprungmöglichkeiten zwischen den Views, festgelegt.

9.3. Der Prototyp proFOUND-Manager

9.3.1. Architektur

Im folgenden wird anhand eines Anwendungsbeispiels der Prototyp des proFOUND-Manager-Projekts[1] dargestellt, in dem versucht wurde, die zuvor beschriebenen Konzepte in ein rechnergestütztes Werkzeug umzusetzen. Abb. 9-3 zeigt die Komponenten des Gesamtsystems.

Es besteht unter dem Dach des integrativen proFOUND-Manager aus dem gemeinsamen Basismodul proFOUND-Base zur objektorientierten Datenhaltung, den beiden Funktionsmo-

[1] Der hier beschriebene Prototyp basiert auf Vorarbeiten in Form von Projekt-Lehrveranstaltungen, Studien- und Diplomarbeiten sowie integrativen Erweiterungen des Autors an der TU-Berlin (vgl. [Mülheims 92, proFound-Manager]).

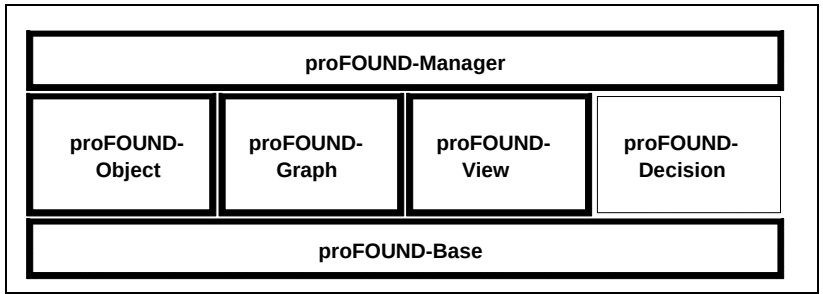


Abb. 9-3: Die Modulstruktur des proFOUND-Systems

dulen proFOUND-View zur individuellen Datenanalyse und proFOUND-Decision zur Unterstützung der Generierung und Bewertung von Handlungsalternativen sowie den (alternativen) Entwickler-Komponenten proFOUND-Object und proFOUND-Graph zur Spezifikation und Verwaltung von Informationsobjekten. proFOUND-View und proFOUND-Decision gestatten neben dem direkten Zugriff auf die Datenbasis insbesondere die Konfiguration daten- und funktionsmäßig spezialisierter Anwendungssichten, die vom Anwender individuell und je nach Bedarf geladen werden können.

Im einzelnen übernehmen die Komponenten folgende Aufgaben:

- *Der proFOUND-Manager*
  Von hier lassen sich die verschiedenen Komponenten des proFOUND-Manager-Systems aufrufen. Damit die Programmteile untereinander Objektstrukturen austauschen können, stellt der proFOUND-Manager eine eigene Objektzwischenablage zur Verfügung, die aus allen "proFOUND"-Anwendungen gestartet werden kann. Das proFOUND-Manager Modul bildet die Minimalkonfiguration, die durch weitere Anwendungsmodule ergänzt werden kann.

- *Die Datenverwaltung proFOUND-Base*
  Das Modul "proFOUND-Base" realisiert die gesamte Datenverwaltung. Diese Komponente ermöglicht die Konstruktion, Verwaltung und Darstellung komplexer Datenstrukturen, wobei diese optional auch außerhalb der "proFOUND-Umgebung" gespeichert werden können. Das Konzept der Datenbasis mußte für eine erste prototypische Realisierung in einigen Teilen stark vereinfacht werden. Persistente Datentypen im strengen Sinne, Netzwerkfähigkeit und Verteiltheit wurden nicht realisiert. Es ist jedoch möglich, ein Objekt, in seine Bestandteile zerlegt, mitsamt seinen Unterobjekten, in einer eigenen Datei zu speichern und von dort wieder zu laden. Diese Möglichkeit, Objekte von jedem

Modul aus speichern und wieder laden zu können, stellte eine Mindestanforderung an das System dar. Damit ist es möglich, Objekte, auch von anderen Applikationen aus zu laden, die nicht zum proFOUND-Manager System gehören. In einer späteren Ausbaustufe wird diese Hilfskondtruktion durch eine Repository-Standardsoftware, vorzugsweise mit objektorientiertem Datenbankanschluß für das Informationsmodell zu ersetzen sein.

Der objektorientierte Ansatz ermöglicht es, den Grundbausteinen Funktionalität mitzugeben. Das heißt, jedes in der Datenbasis gespeicherte Objekt verfügt über die zu seiner Manipulation notwendigen Operationen selbst (Prinzip der Datenkapselung). So verfügt jedes Objekt selbst über die Fähigkeit, seine inhaltliche Struktur in einer Datei zu speichern und aus dieser wieder zu laden. Darüber hinaus besteht die Möglichkeit klassenspezifischer Methoden zur Darstellung oder Weiterverarbeitung, z.B. für analytische Auswertungen.

❒ *Die Objekteditoren proFOUND-Object und proFOUND-Graphic*
In der Komponente "proFOUND-Object" sind viele spezialisierte Werkzeuge zur Erstellung, Bearbeitung und Repräsentation von proFOUND-Base-Datenstrukturen unter einem proFOUND-Object-Fenster zusammengefaßt. proFOUND-Object wird durch das Modul "proFOUND-Graphic" ergänzt, das auch eine graphische, Bearbeitung und Darstellung von proFOUND-Base-Strukturen erlaubt. Insofern repräsentieren diese Komponenten den Objekt-Konfigurator des Vorgehensmodells.

❒ *Die EIS-Komponente "proFOUND-View"*
Die Komponente "proFOUND-View" ermöglicht es, auf der Basis der mit proFOUND-Object erzeugten Elemente eine graphische Repräsentation und Bearbeitung ausgesuchter Daten im Rahmen eines unternehmensspezifischen Berichtwesens in sogenannten Views (Sichten) zusammenzufassen und diese in der Datenbasis zu speichern. Insofern repräsentiert sie erste Teile des Applikations-Konfigurators des Vorgehensmodells.

Die View-Vernetzung ist im Prototyp noch auf die Basis-Struktur im Objektmodell beschränkt. In einer weiteren Ausbaustufe wird eine zusätzliche Konfigurator-Komponente auf Basis der Hypermedia-Technologie die Erstellung alternativer Informationsnetzwerke übernehmen, die dann in Form von attributiven Eintragungen in den Informationsobjekten realisiert werden soll. Die entsprechenden Voraussetzungen sind mit Hilfe des Objekt-Konfigurators auf Ebene der Basis-Objektklassen vorzunehmen.

❒ *Die DSS-Komponente "proFOUND-Decision"*
Die Komponente "proFOUND-Decision" wurde noch nicht realisiert. Prinzipiell wird die Aufgabe von "proFOUND-Decision" darin bestehen, auf der Grundlage der mit "proFOUND-View" selektierten, verdichteten bzw. Basis-Daten strategische Entscheidungen im Rahmen der für ein Unternehmen wichtigen Planungen und Zielsetzungen zu unterstützen. Hierfür wurden Schnittstellen bei der Konzeption berücksichtigt, sowohl zu proFOUND-Methoden als insbesondere auch zu externen, spezialisierten Standard-Software-Systemen..

9.3.2. Beispiel-Szenario

Im folgenden wird die Funktionsweise des proFOUND-Systems ausschnittsweise an folgendem Beispiel-Szenario aus einem typischen Problemfeld von Führungsinformationssystemen, der Abweichungsanalyse demonstriert:

Ob eine berichtete, positive aggregierte Umsatzentwicklung wirklich beruhigend ist, hängt z.B. davon ab, daß nicht negative Entwicklungen bestimmter Produkte und Regionen durch andere, positive überkompensiert werden. Entsprechende Hinweise müssen also automatisch vom Berichtswesen generiert oder durch entsprechende (textuelle) Kommentierungen der Referenten mitgeliefert werden. Ob solche negative Detailentwicklungen kritisch sind, ist selten eine rein numerische Frage. Als mögliche Ursachen kommen bspw. ein geplanter Produktwechsel, Lieferengpässe, verändertes Käuferverhalten oder verstärkte Konkurrenz in Betracht. Auch für die Ableitung geeigneter Gegenmaßnahmen spielen solche Zusatzinformationen über das Umfeld eine entscheidende Rolle. Für den Fall des Produktwechsels soll im folgenden untersucht werden, welche strukturellen und funktionalen Anforderungen sich daraus an ein effektives Berichtswesen ergeben können:

Über den entsprechenden (textuellen) Hinweis des verantwortlichen Produkt(gruppen)-Leiters hinaus könnte der Manager Wert darauf legen zu erfahren, ob bzw. wie das Timing der Umstellung funktioniert, d.h. wie sich die Summe von Auslauf- und Nachfolgeprodukt verhält. Hierzu müßte eine weitere Detailgrafik, evtl. ebenfalls mit zusätzlichen textuellen Kommentaren abrufbar sein. Sofern nicht bereits in der Berichtsstruktur vorgesehen, ist eine entsprechende Handlungsaufforderung an den zuständigen Produktmanager oder Referenten abzusetzen. Hierzu sowie für weitergehende Rückfragen müssen diese verantwortlichen Ansprechpartner aus dem aktuellen Kontext des Berichtswesens identifizierbar und am besten direkt ansprechbar sein, z.B. indem automatisch eine Verbindung per Telefon oder EMail hergestellt wird. Diese sollten dann möglichst einfach die Berichtsstruktur ändern bzw. erweitern können, z.B. indem die bereits einzeln im Berichtswesen enthaltenen Produkte zu einer neuen, Analyse-technischen Produktgruppe zusammengefaßt und eingefügt werden, ohne daß erneut die Spezifikation der Datenherkunft, -berechnung und Darstellungsform wiederholt werden müssen.

9.3.3. Umsetzung des Beispiel-Szenario

Basis für alle späteren Anwendungsfunktionen bildet die (einmalige) vollständig objektorientierte Spezifikation der Datenbasis. Abb. 9-4 zeigt einen Ausschnitt aus dem Spezifikationsdialog mit Hilfe der beiden Objekteditoren proFOUND-Object und proFOUND-Graph. Im "hinteren" Teil ist die grafische Repräsentation des generierten Informationsobjekts CSF (critical success factors) dargestellt. Es besteht im Beispiel erst aus dem zusammengesetzten Objekt Umsatz, für das seinerseits erst eine Produktgruppe PG-I definiert ist. Diese setzt sich ihrerseits aus drei Produkt-Objekten zusammen, deren Grundstruktur mit der Typ-Komponente *Zeitreihe* als Quartals-Untergliederung generiert und danach individuell erweitert wurde. Im Falle von Produkt-A wurde aus der Datenbasis mit Hilfe der Bibliotheks-Funktion ein vorgefertigtes Produkt-Info-Objekt, im Falle von Produkt-B ein elementares Objekt "Kommentar" vom Typ Text angehängt.

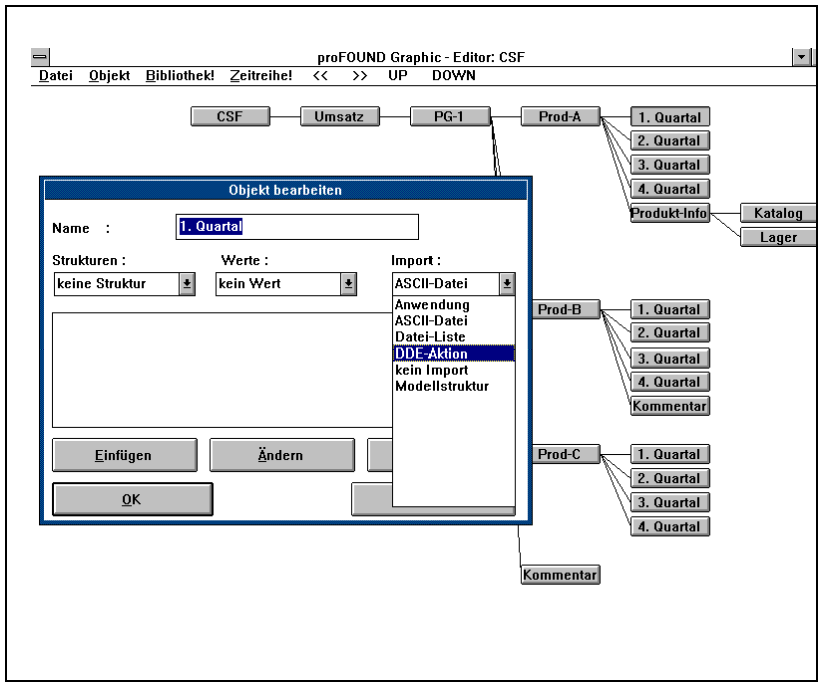


Abb. 9-4:	Spezifikation von Informations-Objekten

In der Abbildung ist exemplarisch die Spezifikation der Datenselektion für den Umsatz des Produktes A im 1. Quartal dargestellt. Alternativ zu - in der Praxis eher untypischen - Direkteingaben von Wert und Typ sind hier Schnittstellen zu externen Datenbeständen, z.B. einer SQL-Datenbank vorgesehen, die neben einer dynamischen Wertaktualisierung auch eine automatische Typ-Anpassung vornehmen sollen. Schließlich sehen die Import-Alternativen auch das Laden komplexerer Unterstrukturen, z.B. aus mehrdimensionalen Planungsmodellen (Import Modellstruktur) vor.

Im weiteren Verlauf der Objekt-Spezifikation können weitere Kennzahlen-Objekte hinzugefügt werden, wobei Kopier- und Einfüge-Operationen auf bereits definierte Strukturen zum Einsatz kommen können.

Dadurch, daß eingefügte Objekte als Instanzen elementarer Klassen gebildet werden, besitzen sie bereits die notwendigen Informationen, wie sie Daten aus welchen Quellen extrahieren müssen und wie diese in späteren Abfragen darstellbar sind. Dies gilt auch für zusammengesetzte Objekte, die sämtlich auf elementare Objekttypen zurückzuverfolgen sind. Besteht bspw. ein zusammengesetztes Objekt wie Produkt-A aus mehreren numerischen Elementar-Objekten, hier die Quartale, so kann es daraus automatisch ableiten, daß neben einer tabellarischen Form auch Grafiken in Frage kommen. Welche Grafiken geeignet sind, kann wiederum aus den Wert-Ausprägungen abgeleitet werden. Wären bspw. positive und negative Werte vorhanden, so scheidet ein Kuchendiagramm sofort aus. Liegt ein Elementar-Objekt vom Typ Text vor, wie bspw. im Falle der Kommentar-Objekte, kann automatisch ein Textfenster, wahlweise mit oder ohne Editierfunktion generiert werden. Alle diese Funktionalitäten sind bei den Klassen der Elementar- bzw. zusammengesetzten Objekte zentral hinterlegt und werden durch die Objekt-Instanziierung auf diese "vererbt". Änderungen der Funktionen (an den Klassen) erstrecken sich folglich automatisch auf alle Objekte und damit auch alle Anwendungen im Unternehmen.

Diese Funktionsweise kann im zweiten Schritt der Anwendungsentwicklung, der Spezifikation einer Teilsicht bspw. für den Produktmanager der Produktgruppen PG-1 und 2 dargestellt werden. Abb. 9-5 zeigt einen Ausschnitt aus dem Dialog mit der proFOUND-View-Komponente. Hierzu wurde das zuvor spezifizierte Informations-Objekt CSF geladen.

Hierbei können beliebige Teilmengen aus der Objektstruktur selektiert werden, bspw. die (aggregierten) Umsatz-Objekte für die Produktgruppen PG-1 und 2. Zur Unterstützung der Übersichtlichkeit lassen sich nicht interessierende Hierarchiestufen, hier die Stufen 4 und 5, ausblenden. Aufgrund der Typen der selektierten Objekte, die sich bei Bedarf auch rekursiv aus den Sub-Objekten ableiten, werden dem Entwickler in Frage kommende Darstellungs-

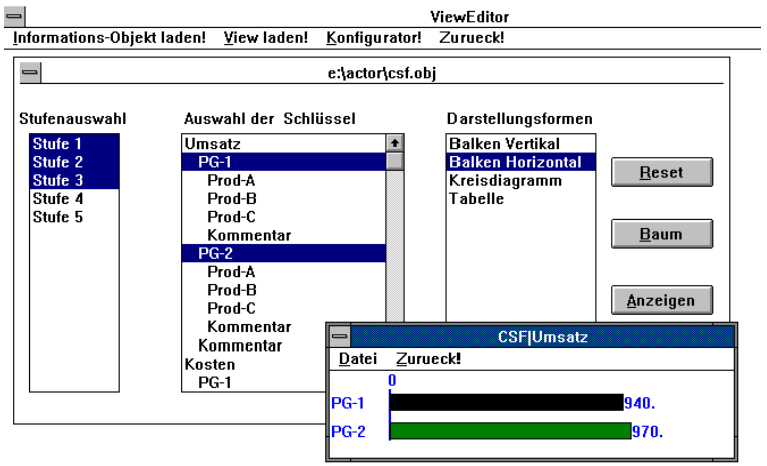


Abb. 9-5: Spezifikation einer Teilsicht

formen in der rechten Auswahlbox angeboten. Von diesen kann er eine für die Standard-Darstellung des zu generierenden View-Fensters auswählen und sich das Objekt anzeigen lassen. Dieses View-Element läßt sich dann seinerseits als Objekt sichern. Auf diese Art und Weise können alle gewünschten Teilsichten spezifiziert werden. Insbesondere können dabei verschiedenste Informationsobjekte und damit Informationsquellen zum Einsatz kommen.

In einem weiteren Schritt, der Konfiguration, können nun individuelle EIS-Anwendungen, z.B. für unterschiedliche Anwender oder Anlässe, wie Produktmanager-Sitzung, Vorstandssitzung, wöchentliche oder tägliche Briefings usw., aus diesen View-Elementen zusammengesetzt werden. Auch diese aus Einzel-Views zusammengesetzten EIS-Anwendungen lassen sich in der Objekt-Datenbasis abspeichern und können von (autorisierten) Anwendern bei Bedarf geladen werden. Abbildung 9-6 zeigt eine solche (konfigurierte) Anwendung, bestehend aus einer geöffneten und einer weiteren verfügbaren, momentan geschlossenen EIS-Anwendung.

Die geöffnete EIS-Anwendung-I (Umsätze PG-I/II) enthält zunächst nur den zuvor spezifizierten Teil-View über die Umsätze der PG-1 und 2 sowie einen optionalen, noch geschlossenen, weil leeren Kommentar-View. Dieser kann geöffnet, editiert und gespeichert werden.

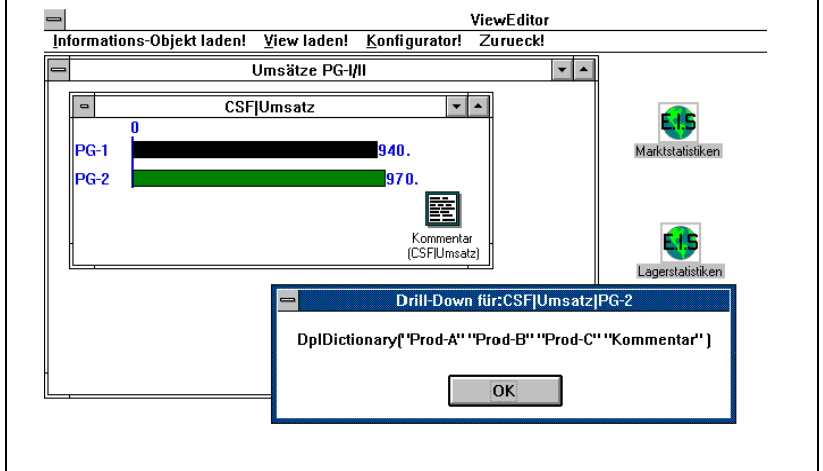


Abb. 9-6: EIS-Anwendung vor Drill-Down und Kommentierung

Damit steht er sofort allen anderen Anwendern zur Verfügung, die dasselbe Informations-Objekt (Umsatz) zu ihrer Sicht zählen. Ferner verfügt der Grafik-View aufgrund seines aggregierten Charakters über eine Drill-Down-Funktionalität, die durch das Anklicken bspw. des PG-2-Balkens in der Grafik aktiviert werden kann. Das System ermittelt daraufhin aus der Sub-Struktur des Informations-Objekts, welche abhängigen Views zur Verfügung stehen und bietet sie dem Anwender zur Auswahl an (vgl. prototypischen Hinweis in Abb. 9-6). Nach Auswahl und Bestätigung werden die entsprechenden Views entsprechend ihrem Typ automatisch generiert. Das Ergebnis ist in Abb. 9-7 zusammen mit den aufgrund der Analyse dieser und weiterer Detailgrafiken für die Produkte B und C editierten Kommentar-Views dargestellt.

Darüber hinaus können den View-Elementen weitere Funktionalitäten mitgegeben werden, die bspw. durch eine dynamische Menüleiste oder - um den Bildschirm nicht zu überladen - per Mausklick in den View abgerufen werden können. Dazu zählen etwa ein Drill-up als Gegenstück zu Detaillierung, die Spezifikation von Grenzwerten für ein Exception-Reporting, die Ergänzung der Grafik um Durchschnitts- und Trendberechnungen oder einfach das Wechseln der Darstellungsform in andere, wieder vom System vorzuschlagende Grafiktypen.

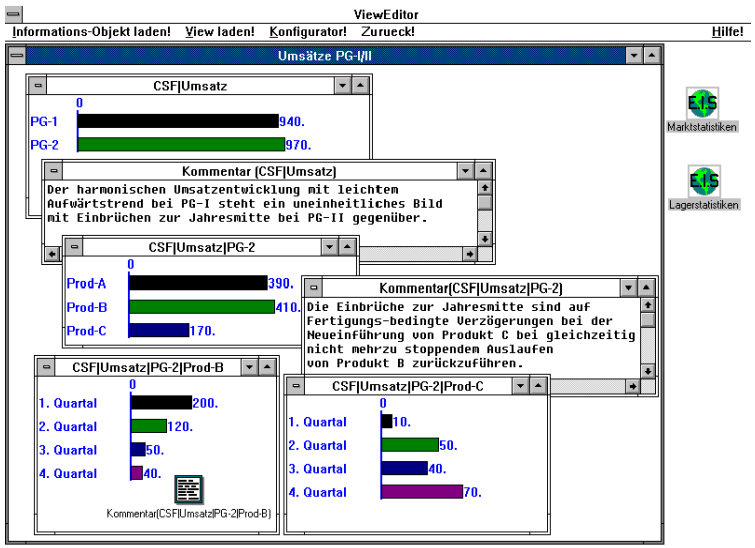


Abb. 9-7:	EIS-Anwendung nach Drill-Down und Kommentierung

Die Anforderung weiterer, noch nicht vorgesehener Objektstrukturen, z.B. die Analyse von zwei sich ablösenden Produkten läßt sich prinzipiell auf folgende Arten realisieren, je nachdem wieviel "EDV-Engagement" dem Manager zu eigen ist bzw. zugemutet werden kann:

- Die direkteste, aber auch für den Anwender "arbeits-intensivste" Form wäre die Öffnung des Zugangs zu Objekt- und View-Editor, um sich selbst das entsprechende Objekt zu konfigurieren.
- Die einfachste Form für den Manager könnte darin bestehen, über eine Kommunikationskomponente, die durch das Ansprechpartner-Objekt aktiviert wird, dem Verantwortlichen (telefonisch und damit persönlich oder per EMail und damit unabhängig von dessen aktueller Erreichbarkeit) das Anliegen zu delegieren.
- Dazwischen liegen Fälle, die über eine entsprechend erweiterte View-Funktionalität das Selektieren und Kombinieren von Informationsobjekten verschiedener Views zu neuen, temporären Views erlauben.

Bei der Entscheidung für eine Alternative sind außer Aspekten der Dauerhaftigkeit solcher Anfragen insbesondere auch Aspekte der Unternehmenskultur, Mitarbeiter-Motivation usw. zu berücksichtigen. Im Zweifel ist eher die kommunikative, delegierende Lösung zu bevorzugen, die alle Unternehmensebenen zu aktiven Bestandteilen des Informationssystems macht. Der dadurch erhoffte und erwartbare Beitrag zur Steigerung der Akzeptanz von Informationssystemen stellt bei aller Eleganz und Effizienz rechnergestützter Lösungen nach wie vor den vielleicht wichtigsten kritischen Erfolgsfaktor für Unternehmen dar.

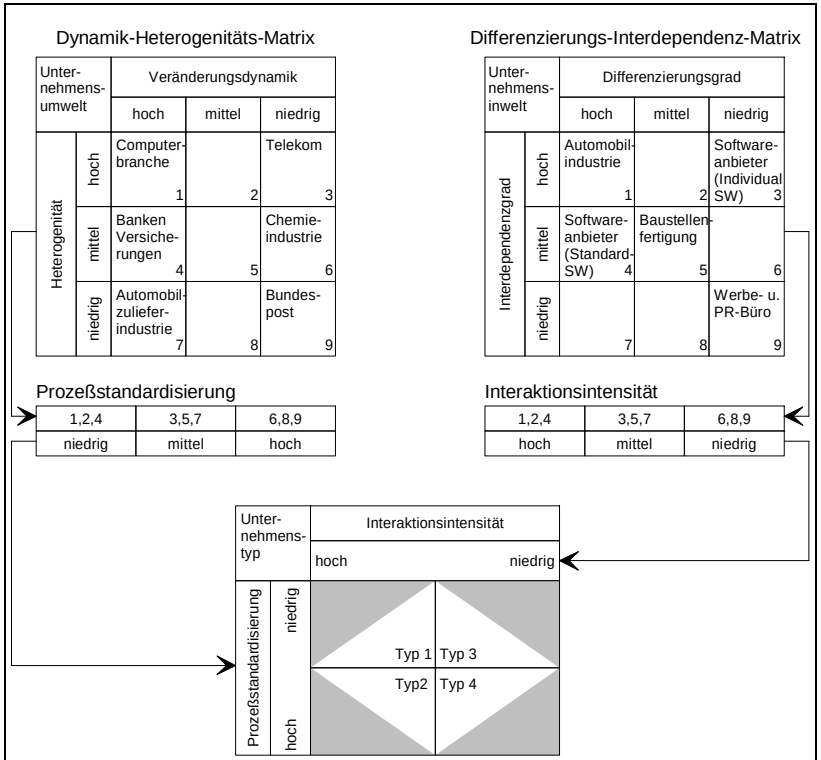
10. Perspektiven und Ausblick

10.1. Konsequenzen für die Unternehmens-Organisation

Die dargestellten Konzepte, Informationstechnologien und Software-Konstrukte bzw. Bausteine tragen allesamt das Potential einer schnelleren, zielgerichteten Bereitstellung relevanter Informationen unmittelbar am Arbeitsplatz von Entscheidungsträgern in sich. Nüchtern betrachtet können sie aber im Sinne der Differenzierung von Wagner bei der Zieldefinition von Decision Support Systemen zwischen Effizienz und Effektivität (vgl. [Wagner 80, DSS], S. 214) nur einen (technischen) Beitrag in den Termini des "schneller" und "genauer" leisten. Zahn drückt dies wiefolgt aus (vgl. [Greschner 92, Erfolgsfaktor Information], S. 14): *"Allein das Vorhandensein von Information ist kein Garant für die Entwicklung eines Vorsprungs zur Konkurrenz."* Erst *"die Fähigkeit, aus der Verarbeitung von Daten entstandene Informationen zu nutzen, also neues Wissen zu generieren und zielgerichtet zu verwenden, ist die entscheidende Determinante des Erfolgs."*

Dieser (permanente) Transformationsprozeß ist eng verknüpft mit dem installierten bzw. praktizierten Führungs- und Entscheidungssystem eines Unternehmens, und damit der Aufbau- und Ablauforganisation. Der Wirkungsgrad neuer Technologien der in dieser Arbeit beschriebenen Art hängt entscheidend von parallelen, organisatorischen Anpassungen ab. Folgerichtig stellt auch Bullinger fest, daß mehr notwendig ist als eine *"Elektrifizierung des Vorstandsberichtswesens"*. Erst eine Art *"Palastrevolution, die primär auf eine organisatorische Umwälzung des Management- bzw. Führungssystems gerichtet ist"*, kann zu einer spürbaren Entfaltung der Nutzenpotentiale neuer Informationstechnologien führen und auch synergetische Effekte bewirken. *"Flache Unternehmenshierarchien, Macht- und Kompetenzumverteilungen zwischen Linie und Prozeß sowie Downsizing, Outsourcing und Dezentralisierung"* sind notwendige Instrumente zur Realisierung der Leitlinie "Lean Management" auch im administrativen Bereich (vgl. [Bullinger 92, Chefinformationssysteme], S. 49-50).

So wie technische Features erst im passenden, organisatorischen Umfeld wirksam werden können, machen sie nicht in jedem Unternehmen gleichermaßen Sinn. Vielmehr müssen die geeigneten Technologien entsprechend der Unternehmens- und Umwelt-, d.h. Markt-Charakteristika individuell bestimmt werden. Bullinger schlägt hierfür die in Abb. 10-1 skizzierte Planungsmethode vor (vgl. [Bullinger 92, Chefinformationssysteme], S. 57).



nach: [Bullinger 92, Chefinformationssysteme], S. 57

Abb. 10-1: EIS-Planung Phase 1: Bestimmung des Unternehmenstyps

In einem dreistufigen Prozeß wird mit Hilfe der Portfolio-Technik aus externen und internen Charakteristika eine Typisierung von Unternehmen abgeleitet, die sich an Gestaltungsanforderungen für die technische und organisatorische Ausstattung des Führungssystems orientiert. Bestimmungsgrößen der Unternehmensumwelt sind dabei Veränderungsdynamik (der Wettbewerber) und Heterogenität der (Produktpalette), die zum Merkmal Prozeßstandardisierung aggregiert werden. Unternehmensinterne Kriterien sind Interdependenzgrad (aufgrund von Aufgabenteilung) und Differenzierungsgrad (im Hinblick auf "Einzelfertigung" aufgrund individueller Kundenwünsche), die zum Merkmal Interaktionsintensität aggregiert werden.

Auf der Basis dieser Typisierung läßt sich dann eine Zuordnung adäquater, abgestimmter Maßnahmen der Gestaltungsdimensionen personell, technokratisch, organisatorisch und (schließlich) informationell vornehmen (vgl. Abb 10-2).

Gestaltungsdimensionen \ Unternehmenstyp	Typ 1 interaktionsintensiv, wenig standardisiert	Typ 2 interaktionsintensiv, standardisiert	Typ 3 interaktionsarm, wenig standardisiert	Typ 4 interaktionsarm, standardisiert
personelle Maßnahmen (Instrumente)	• kooperativer Führungsstil • MbO • Intrapreneurship • Erfolgsbeteiligung • Förderung von –Selbstkontrolle –Sozialkompetenz	• Kultur • Delegation • Leistungsstandards • Förderung der –Fachkompetenz der Mitarbeiter –Sozialkompetenz der Führungskräfte	• Förderung der Fachkompetenz • cross-staffing (Bündelung der Fähigkeits-potentiale) • Partizipation • Spezialistentum	• Spezialisierung (im tayloristischen Sinn) • Führungserfahrung • autoritärer Führungsstil • Laufbahnkonzepte nach Senioritätsprinzip
technokratische Maßnahmen (Instrumente)	• KAIZEN • Center-Konzepte • Projekt- und Prozeß-kostenrechnung • Target-Management • KEIRETSU • KANBAN	• Kosten- und Leistungsrechnung • Controlling • Normierung/ Standardisierung • strategische und operative Planung	• Wertanalyse • TQM • Auslastungsplanung • Projektplanung und -kontrolle • Multi-Projecting	• Budgetierung • ausgeprägtes Planungssystem • Controlling (Kostenkontrolle) • Standards/Normen • Qualitätskontrolle
organisatorische Maßnahmen (Instrumente)	• Projektorganisation • Teamorganisation • Prozeßorganisation • KASOKU • Selbstorganisation	• hierarchische Strukturen • Kollegienorganisation • Rationalisierung beste-hender Abläufe • Segmentierung	• Projekt-/Team-organisation • Simultaneous Engineering • F&E-Netzwerke • Matrixorganisation	• funktionale Organisation • Stabsorganisation
informationelle Maßnahmen (Instrumente)	• CIS mit Betonung auf –Adhoc-Abfragen –DSS-Komponenten –externe Information –Querschnitts-information –unternehmens-übergreifendes CIS	• CIS mit Betonung auf –E-Mail –Standardberichte –interne Information –Querschnitts-information –unternehmens-weites CIS	• CIS mit Betonung auf –Projektmanage-mentsystem –DSS-Komponenten für Simulation und Prognose –Adhoc-Abfragen –bereichsweites CIS	• CIS mit Betonung auf –Standardberichte –funktionsbezogene Detailinformationen –bereichsweites CIS

Quelle: [Bullinger 92, Chefinformations-systeme], S. 58

Komplexität und strategische Wichtigkeit des CIS-Projekts

Wirkung isolierter informationeller Maßnahmen

Abb. 10-2: EIS-Planung Phase 2: Maßnahmenauswahl je Unternehmenstyp

Je weniger standardisiert bzw. statisch Leistungsangebot und Märkte, auf denen das Unternehmen agiert, und je geringer Variantenreichtum der Produkte und Grad der Arbeitsteilung, intern wie extern, umso größer sind (erwartungsgemäß) Bedarf und qualitative Anforderungen an ein unternehmensweites Führungsinformationssystem. Im Kontext einer zunehmenden informationstechnischen Vernetzung mit Zulieferern, z.B. in der Automobilindustrie, unter dem Schlagwort der zwischenbetrieblichen Integration erweitern sich die Anforderungen auf unternehmensübergreifende EIS (CIS).

Einen strategie-orientierten Ansatz zur Abstimmung von technologischem und organisatorischem Wandel schlägt Zahn vor (vgl. [Greschner 92, Erfolgsfaktor Information], S. 13). Danach sind Planung, Umsetzung und Kontrolle des Einsatzes von Informationstechnologien (IT-Strategieplanung) als integraler Bestandteil in den Planungszyklus aller Strategie-

komponenten eines Unternehmens einzubinden. IT-Aussagen müssen bereits bei der Festlegung der Geschäftsfelder bis hinauf zur Unternehmensvision Berücksichtigung finden. Abb. 10-3 illustriert diesen system- und regelkreis-orientierten Ansatz (vgl. [Zahn 90, Wettbewerbsfaktor Information], S. 497).

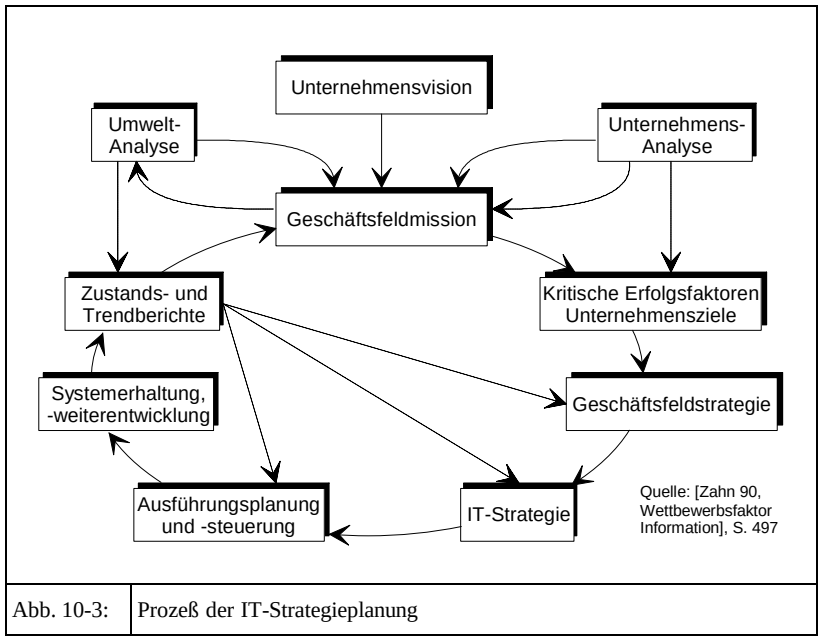


Abb. 10-3: Prozeß der IT-Strategieplanung

Informationstechnologie und Information selbst erweitert sich damit von der Ebene einer Dienstleistung bzw. eines Mittels zum Zweck hin zur Ebene eines eigenständigen Produkts. Informationstechnologie, ursprünglich zur Optimierung der Leistungserstellung eingesetzt, wird zum Gegenstand einer aufwertenden Ergänzung der bisherigen Produkte. In letzter Konsequenz können daraus neue Unternehmenssparten mit eigenständigen Produkten entstehen, wodurch ein Wandel der bearbeiteten Geschäftsfelder eingeleitet wird.

Zur Aktivierung dieser Strukturen und Prozesse sind Änderungen im Führungs- und Anreizsystem notwendig. Eigenverantwortliches, unternehmerisches Denken und Handeln in Abteilungen (Intrapreneurship) unter dem Dach des Beitrags zum Gesamtunternehmenserfolg im Sinne von Profit-Centers sind erforderlich. Alle weiteren personellen Maßnahmen, wie

Erfolgsbeteiligungen, Management by Objectives (MbO) oder Team- und Projektorganisation müssen auf die Etablierung eines kooperativen Führungsstils gerichtet sein.

10.2. Konsequenzen für die Personal-Qualifikation

Von diesen weitreichenden organisatorischen Umgestaltungen leiten sich die wesentlichen Konsequenzen für die Qualifikation des (Führungs-)Personals ab. Weniger technische Fertigkeiten im Umgang mit Rechnern und Softwarewerkzeugen stellen die kritischen Erfolgsfaktoren dar, denn Bereitschaft und Fähigkeit zur sozialen Kommunikation und Kooperation. Die in der Arbeit beschriebenen Fortschritte bezüglich der Bedienerfreundlichkeit für Anwender und Entwickler von Softwaresystemen verstärken diese Verlagerungstendenz der Qualifikationsanforderungen eindringlich. Daß dies allein jedoch nicht ausreicht, daß vielmehr eine allzu starke Technikorientierung bei der Gestaltung und Einführung von Informations- und Kommunikationstechnologien zu kontraproduktiven Effekten führen kann, belegt Bainbridge eindrucksvoll mit dem Schlagwort "Paradoxien der Automatisierung" (vgl. [Bainbridge 83, Ironies of automation]): *"Verlust an Qualifikationen durch den Einsatz hochtechnisierter Systeme, obwohl deren Bedienung - zumindest im Problemfall - genau diese Qualifikationen erfordert."*

In Anlehnung an die Unterscheidung von Ulich zwischen technik- und arbeitsorientierter Perspektive beim Einsatz neuer Technologien (vgl. [Ulich 91, Arbeitspsychologie], S. 215 ff.) fordern Dunckel und Volpert folgerichtig aus Sicht der Arbeits- und Organisationspsychologie eine stärkere Betonung der arbeitsorientierten Seite. Ziel dabei ist es Bedingungen zu schaffen, die eine Entfaltung von Motivation und Kreativität ermöglichen (vgl. [Dunckel 92, Kontrastive Aufgabenanalyse], S. 207). Zur operationalen Umsetzung schlagen sie einen Leitfaden zur Kontrastiven Aufgabenanalyse und -gestaltung (KABA) vor, der die Konfiguration von Arbeitsaufgaben an sogenannten Humankriterien ausrichtet (vgl. [Dunckel 92, Kontrastive Aufgabenanalyse], S. 211):

- ❒ großer Entscheidungsspielraum
- ❒ angemessener zeitlicher Spielraum
- ❒ Durchschaubarkeit und Gestaltbarkeit gemäß eigener Ziele
- ❒ keine Behinderung der Aufgabenerfüllung durch organisatorische oder technische Bedingungen
- ❒ ausreichende körperliche Aktivität
- ❒ konkreter Kontakt zu materiellen und sozialen Bedingungen des Arbeitshandelns
- ❒ Beanspruchung vielfältiger Sinnesqualitäten
- ❒ Variationsmöglichkeiten bei der Erledigung der Arbeitsaufgaben
- ❒ unmittelbare zwischenmenschliche Kontakte.

Dieser Ansatz kann einen pragmatischen Beitrag dazu leisten, dem im Rahmen der Software-Ergonomie diskutierten Primat der Arbeitsaufgabengestaltung durch methodische Fundierung zum Durchbruch zu verhelfen und damit die bisher beklagte allzu starke Konzentration auf die Analyse und Gestaltung von Benutzeroberflächen zu relativieren (vgl. [Bullinger 87, Software-Ergonomie]; [Hampe-Neteler 92, Software-Ergonomie]). Gleichzeitig erhöht er natürlich die Qualifikationsanforderungen an das Entwicklungspersonal erheblich, da zunehmend ein interdisziplinäres Profil gefordert ist.

Ein Ansatz zur Milderung dieses Problems, das gleichermaßen für den adäquaten Einsatz innovativer Software-Konstrukte, z.B. der Objektorientierung oder Datenmodellierung, gilt, kann darin bestehen, Methoden und Techniken durch Integration in rechnergestützte Entwicklungswerkzeuge zu operationalisieren. Für den Bereich der strukturierten Programmierung wurde dieses Konzept in Form dedizierter Programmiersprachen, z.B. Pascal oder Modula, für den Bereich der Objektorientierung z.B. durch Objectworks/Smalltalk, partiell erfolgreich praktiziert, indem methoden-konforme Strukturen (Templates) vorgegeben bzw. methoden-konfliktäre Konstrukte (GOTO) verbannt wurden. Die Bemühungen des computergestützten Software-Engineering (CASE) zur Schließung der Medienbrüche zwischen Analyse, Entwurf und Implementierung bieten hier ein bislang bei weitem noch nicht ausgeschöpftes Feld zur rechnergestützten Anleitung, Führung und Unterstützung von Anwendungsentwicklern in Bezug auf die Berücksichtigung der beschriebenen organisatorischen und humanwissenschaftlichen, für den Erfolg speziell von Führungsinformationssystemen kritischen Faktoren.

10.3. Konsequenzen für standardisierte Entwicklungs-Werkzeuge

Für eine Umsetzung struktureller Konzepte oder methodischer Innovationen in Anwendungssysteme der Praxis kommen grundsätzlich zwei Wege in Betracht:

- Ausbildung und Schulung des Entwicklungs- und Wartungspersonals in Bezug auf Wesen, Vor- und Nachteile sowie adäquate Anwendung der Konzepte, Methoden und Werkzeuge bei Entwurf und Implementierung.
- Implementierung der Konzepte in Form von Software-Bausteinen (Konstrukten) derart, daß für den Anwendungsentwickler hiervon eine "Führung" oder "Anleitung" mit der Folge eines "learning-by-doing" ausgeht.

Erst eine Kombination beider Wege kann einen Wissenstransfer zwischen Theorie und Praxis auf breiter Ebene sicherstellen. Die Anfänge der Anwendungsentwicklung wurden mangels spezialisierter Konstrukte in nennenswertem Umfang nahezu ausschließlich durch den

ersten Weg geprägt, d.h. die Ausbildung von Anwendungsprogrammierern. Da sich Software-Entwicklung zu dieser Zeit mehrheitlich in der EDV-Abteilung vollzog, war dies vertretbar. Mit der Verlagerung des Schwerpunkts der "Programmierung" in die Fachabteilungen gewinnt der zweite Weg zunehmend an Bedeutung, zum einen weil eine Nachschulung kaum machbar und angesichts der mehr fachlichen Orientierung auch nicht sinnvoll erscheint. Zum anderen, weil eine repetitive Implementierung elementarer Basiskonstrukte mehr noch als in einer EDV-Abteilung nicht vertretbar ist.

Vielmehr müssen Entwicklungswerkzeuge in Form vorgefertigter Konstrukte diese "Schulungsfunktion" quasi mit übernehmen. Hierbei reicht es keineswegs aus, eine Vielzahl von parametrisierbaren Bausteinen in das Werkzeug aufzunehmen, die an typischen Aufgaben- und Problemstrukturen orientiert sind, und so für eine höhere Übereinstimmung zwischen rechnergestütztem und mentalem bzw. Unternehmens-Modell zu sorgen. Daß das bloße Angebot neuer Konstrukte nicht zwangsläufig auch zu entsprechend neu bzw- restrukturierten Anwendungen führt bzw. führen muß, wird bspw. eindrucksvoll belegt durch objektorientierte Erweiterungen klassischer Programmiersprachen oder Entwicklungswerkzeuge. So wird der Entwickler aufgrund der fortbestehenden Verfügbarkeit klassischer Konstrukte nur in den seltensten Fällen darauf hingewiesen, geführt, geschweige denn "gezwungen", innovative und produktivere Konstrukte, wie z.B. Klassenkonzept oder Vererbung auch einzusetzen. Ein solcher "sanfter Druck" kann erst erreicht werden, wenn das Entwicklungswerkzeug von Grund auf auf eine neue konzeptionelle Basis gestellt wird.

Zu diesen Basistechnologien sind für den Bereich der Führungsinformationssysteme die Objektorientierung und die Hypertext/Hypermedia-Technologie zu rechnen. Deshalb sind DSS- und EIS-Generatoren, wie bspw. Pilot, die mit klassischen Wirtssprachen wie PL/1 realisiert sind, abzulehnen. Aber auch eine objektorientierte Basis, die zumindest aufgrund des Erscheinungsbildes Produkten wie LightShip oder Forest&Trees zugeschrieben werden kann, ist ungenügend, wenn sie nur auf Seiten des Werkzeug-Entwicklers zum Einsatz kommt. Die Produktivitätsvorteile dieser Technologie können sich erst dann nennenswert und "rechenbar" entfalten, wenn sie auch dem Anwendungsentwickler zur Verfügung stehen. Bspw. genügt es nicht, ein Spektrum vorgefertigter Informationsobjekt-Klassen, im Falle von Forest&Trees Views, anzubieten, ohne auch die Möglichkeit zu schaffen, einzelne Klassen zu modifizieren oder den Klassenbaum individuell zu erweitern.

Eine weitere Anforderung muß hinzukommen. Sie betrifft die Unterstützung des Anwendungsentwicklers beim Entwurf des Systems, d.h. der Auswahl, Parametrisierung und Kombination der Konstrukte. Handbücher und Online-Hilfesysteme beschränken sich zumeist auf die anwendungsunabhängige Beschreibung von Konstrukten, Parametern und

Funktionen. Erst ein zweites Nachschlagewerk neben der alphabetischen Ordnung, das sich an Anwendungsfunktionen orientiert und umfangreiche Querverweise auf potentiell relevante, vor- bzw. nachgelagerte Analyse- oder Auswertungsschritte enthält, kann eine drohende Desorientierung des Anwenders durch eine erhöhte inhaltliche Selbsterklärungsfähigkeit des Entwicklungswerkzeugs vermeiden helfen.

Eine alternative Vorgehensweise ist im Angebot von Musteranwendungen für typische betriebliche Analyse- und Planungsaufgaben zu sehen. Ein Beispiel hierfür stellen die umfangreichen Musterapplikationen des EIS-Generators Pilot dar, die durch die Ergänzung um Parametrisierungskomponenten vom Ansatz her zwar in eine erfolgversprechende Richtung weisen, aufgrund der eingesetzten Basistechnologie jedoch wenig Aussicht auf Erfolg in der Praxis erwarten lassen. Die Attraktivität dieser Lösung für viele, vor allem klein- und mittelständische Unternehmen, geht davon aus, daß damit auch ein Knowhow-Transfer bezüglich Analyse- und Planungs-Methodik verbunden sein kann. Beispiele sind etwa Portfolio-Technik, ABC-Analyse oder Kennzahlensysteme, deren Einführung und Integration in den betrieblichen Planungs- und Kontroll-Prozeß maßgeblich gefördert oder gar initiiert werden kann.

Neben den genannten Basistechnologien, deren Notwendigkeit für alle Typen von Entwicklungswerkzeugen zu fordern ist, ist bei der Ausweitung des Konstruktspektrums bzw. deren Kombination eine starke Orientierung, d.h. Beschränkung, auf methodisch zusammengehörige Anwendungsfelder zu fordern. Das tendenziell methodisch erweiterte Leistungsspektrum von Führungsinformationssystemen sollte nicht dadurch erreicht werden, daß mehr und mehr heterogene Methoden in ein Entwicklungswerkzeug integriert werden, das dadurch in die Nähe eines general-problem-solvers gerückt wird, sondern durch das Prinzip einer klaren Aufgabentrennung. Ein Mehr an Funktionalität ist erfahrungsgemäß zumeist mit einem Weniger an Qualität jeder einzelnen Funktion verbunden. Außerdem resultieren daraus unnötiger Overhead des Entwicklungswerkzeugs, da viele Zusatzfunktionen nur selten benötigt werden, und geringere Übersichtlichkeit für den Anwendungsentwickler.

Dieses Grundprinzip "Tiefe vor Breite" je Entwicklungswerkzeug stellt erhöhte Anforderungen an die Kooperationsfähigkeit, d.h. Offenheit, des Systems, damit das methodisch und inhaltlich heterogene Aufgabenspektrum betrieblicher Arbeitsplätze mit einer adäquaten Rechnerunterstützung versorgt werden kann. Da Kooperation im wesentlichen über die gemeinsame Nutzung bzw. Bearbeitung von Daten, d.h. Informationsobjekten, realisiert wird, verstärkt dieser Aspekt die Tendenz zur Auslagerung der gesamten Daten- resp. Informationsverwaltung in eine eigenständige Server-Komponente. Diese Tendenz wird im Bereich

kommerzieller Anwendungssysteme für operative Unternehmensabläufe bereits erfolgreich praktiziert. Als Beispiel sei nur auf das System R/3 von SAP verwiesen.

In weiterer Konsequenz stellt sich die Auslagerung einer weiteren Komponente zur Diskussion, nämlich der Benutzeroberfläche. Die Realisierung rechnergestützter Arbeitsplätze durch Kooperation von Anwendungssystemen, die jeweils mit spezialisierten Entwicklungswerkzeugen implementiert sind und die allesamt über (umfangreiche) eigene Komponenten zur Gestaltung der Benutzeroberfläche verfügen, kollidiert mit der Anforderung einer einheitlichen Dialogschnittstelle. Zwar haben grafische, Betriebssystem-nahe Oberflächen wie MS-Windows bereits nennenswert zu einer Angleichung und Standardisierung auch der Dialogkonstrukte in den einzelnen Entwicklungswerkzeugen geführt. Das Beispiel des EIS-Generators CommanderEIS zeigt jedoch, daß eine Lauffähigkeit unter MS-Windows noch nicht zwangsläufig auch mit einer Adaption von dessen Dialogprinzipien verbunden sein muß. Unter Effizienzgesichtspunkten, auch aus Sicht der Anbieter von Entwicklungswerkzeugen, ist jedoch eine Tendenz hin zu einer Offenheit auch bezüglich der Benutzerschnittstelle, z.B. eines bzw. mehrerer standardisierter Dialog-Server, bspw. auf der Basis von OSF/Motif, zu befürworten.

Durch eine konzeptionelle Ausrichtung der Anbieter von Entwicklungswerkzeugen an diesen Prinzipien der Basistechnologien sowie der Server-mäßigen Auslagerung von an gemeinsamen Standards orientierter Datenhaltung und Benutzerschnittstelle wird gleichzeitig die Basis für Portabilität geschaffen. Damit können nicht nur zu erwartende Fortschritte im Hardwarebereich, speziell die Verbreitung UNIX-basierter Workstation-Konzepte in die betriebliche Praxis, besser bewältigt werden. Auch können damit erhebliche Produktivitätspotentiale in Bezug auf die Wartung und Weiterentwicklung der angebotenen Entwicklungswerkzeuge erzielt werden. Insgesamt erfordert dies allerdings eine Restrukturierung der Angebotspalette hin zu einem modularen Konzept.

Abbildungsverzeichnis

Literaturverzeichnis

Hinweis: Die Zahlenkombinationen hinter der Kurzreferenz kennzeichnen jeweils Kapitel und Seite der Literaturverweise im Text im Format: Kapitel-fortlaufende Seite.

[Ackoff 67, MIS-Fehler], 3-37
Ackoff, R.E.: Management Misinformation Systems. In: Management Science, December 1967, S. 147-156

[ACM 76, Database-Systems], 8-213
ACM (Hrsg.): Transactions on Database-Systems. No.1, 1976

[Agha 86, Actors], 8-216
Agha, G.: ACTORS: A Model of Concurrent Computation in Distributed Systems, The MIT Press, 1986

[Albayrak 92, Kooperatives Problemlösen], 8-216
Albayrak, S.: Kooperative Lösung der Aufgabe Auftragsdurchsetzung in der Fertigung durch ein Mehr-Agenten-System auf Basis des Blackboard-Modells, Dissertation, TU-Berlin, 1992

[Allen 83, Temporal Intervals], 6-144
Allen, J.: Maintaining Knowledge about Temporal Intervals. In: CACM 26, Nr. 11, 1983, S. 832-843

[Bainbridge 83, Ironies of automation], 10-245
Bainbridge, L.: Ironies of automation. In: [Johannsen 83, man-machine systems], S. 151-157

[Barr 89, Artificial Intelligence], 6-168
Barr, A.; Cohen, P.; Feigenbaum, E. (Hrsg.): Handbook of Artificial Intelligence, Addison-Wesley, 1989

[Behme 93, Führungsinformationssysteme], 3-61
Behme, W.; Schimmelpfeng, K. (Hrsg.): Führungsinformationssysteme - Neue Entwicklungstendenzen im EDV-gestützten Berichtswesen, Gabler, 1993

[Bonczek 81, DSS-Foundations], 6-113
Bonczek, R.H./Holsapple, C./Winston, A.: Foundations of Decision Support Systems, Academic-Press, 1981

[Bond 88, DAI], 8-217
Bond, A.H.; Gasser, L.G. (Hrsg.): Readings in Distributed Artificial Intelligence, Morgan Kaufmann Publ., San Mateo, CA, 1988

[Borgwaldt 80, Modulare Programmsysteme], 6-113
Borgwaldt, H./Höbel, W./Meyder, R./Schlechtendahl, E.G.: Aufbau und Einsatz großer modularer Programmsysteme. In: GI-Fachgruppe MMBS, Heft 3, Oktober 1980, S. 36

[Brause 91, Neuronale Netze], 8-222; 8-224
Brause, R.: Neuronale Netze - eine Einführung in die Neuroinformatik, Teubner, 1991

[Breuker 89, Models of Expertise], 6-140
Breuker, J.; Wielenga, B.: Models of Expertise in Knowledge Acquisition. In: [Guida 89, Topics in Expert Systems], S. 265-295

[Brewka 89, Non Monotonic Logics], 6-160
Brewka, G.: Non Monotonic Logics: an Introductory Overview. In: Tagungsband der KIFS-87, Informatik Fachberichte 202, Springer, 1989

[Buchanan 84, Rule-Based XPS], 6-143
Buchanan, B.; Shortliffe, E. (Hrsg.): Rule-Based Expert Systems, Addison-Wesley, 1984

[Bullinger 87, Software-Ergonomie], 10-246
Bullinger, H.-J.; Fähnrich, K.-P.; Ziegler, J.: Software-Ergonomie - Stand und Entwicklungstendenzen. In: [Schönpflug 87, Software-Ergonomie], S. 17-30

[Bullinger 91, Chefinformationssysteme], 5-101
Bullinger, H.-J./Huber, H./Koll, P.: Chefinformationssysteme (CIS) - Navigationsinstrumente der Unternehmensführung. In: Office Management, 3, 1991, S. 6-20

[Bullinger 92, Chefinformationssysteme], 10-241
Bullinger, H.-J./Koll, P.: Chefinformationssysteme (CIS). In: [Krallmann 92, RAP], S. 49-72

[Bullinger 93, Führungsinformationssysteme], 3-61
Bullinger, H.-J.; Niemeier, J.; Koll, P.: Führungsinformationssysteme (FIS): Einführungskonzepte und Entwicklungspotentiale. In: [Behme 93, Führungsinformationssysteme], Gabler, 1993, S. 44-62

[Carlson 51, Executive Behaviour], 2-23
Carlson, S.: Executive Behaviour: A Study of the Work Load and the Working Methods of Managing Directors, Stockholm: Strömbergs, 1951

[Chandrasekaran 87, Generic Tasks], 6-140
Chandrasekaran, B.: Towards a Functional Architecture for Intelligence Based on Generic Information Processing Tasks. In: Proc. of IJCAI-87, 1987, S. 1183-1192

[Charniak 85, Artificial Intelligence], 6-144
Charniak, E.; McDermott, D.: Introduction to Artificial Intelligence, Addison-Wesley, 1985

[Chen 76, Entity-Relationship Model], 8-213
Chen, P.P.: The Entity-Relationship Model (Towards an Unified View of data). In: [ACM 76, Database-Systems], S. 9-36

[Clancey 83, Framework for Explanation], 6-162
Clance, W.: The Epistemology of a Rule-Based Expert System - a Framework for Explanation. In: AI-Journal 15, 1983, S. 215-251

[Clancey 85, Heuristic Classification], 6-140; 6-164
Clancey, W.: Heuristic Classification. In: AI Journal 27, 1985, S. 289-350

[Conklin 87, Hypertext], 8-218
Conklin, J.: A Survey of Hypertext, MCC Technical Report, No. STP-356-86, 1987

[Corkill 87, distributed problem solving], 8-215
Corkill, D.; Lesser, V.: Distributed Problem Solving. In: [Shapiro 87, Artificial Intelligence], S. 245-251

[Cunis 89, Construction], 6-168
Cunis, R.; Günter, A.; Syska, I.; Bode, H.; Peters, H.: PLAKON - An Approach to Domain-Independent Construction. In: Proc. 2. IEA/AIE, Tennessee, 1989

[Dale 72, Management], 2-22
Dale, E.: Management - Theorie und Praxis der modernen Unternehmensführung, Econ, 1. Aufl., 1972

[Dean 87, Temporal DB-Management], 6-144
Dean, T.; McDermott, D.: Temporal Data Base Management. In: AI-Journal 32, 1987, S. 1-57

[DEC 90, ATIS], 8-214
Digital Equipment Co.: ATIS - A Tool Integration Standard, X3H4/89-108, 1990

[deKleer 86, ATMS], 6-160
deKleer, J.: An Assumption Based TMS. In: AI-Journal 28, 1986, S. 127-162

[Dorffner 91, Konnektionismus], 8-222; 8-224
Dorffner, G.: Konnektionismus - Von neuronalen Netzwerken zu einer "natürlichen" KI, Teubner, 1991

[Doyle 79, TMS], 6-160
Doyle, J.: A Truth Maintenance System. In: AI Journal 12, 1979, S. 231-272

[Drucker 67, Ideale Führungskraft], 2-22
Drucker, P.F.: Die ideale Führungskraft, Econ, 1. Aufl., 1967

[Drucker 77, Management], 2-22
Drucker, P.F.: Management, Pan, 1977

[Dunckel 92, Kontrastive Aufgabenanalyse], 10-245
Dunckel, H.; Volpert, W.: Kontrastive Aufgabenanalyse im Rahmen der Systemgestaltung. In: [Krallmann 92, RAP], S. 205-220

[Durfee 89, Distributed Problem Solving], 8-215
Durfee, E.; Lesser, V.; Corkill, D.: Trends in Cooperative Problem Solving. In: IEEE Transactions on Knowledge and Data Engineering, Vol. 1, No. 1, 1989, S. 63-83

[Ehrenberg 90, Wissensbasierte Systeme], 6-127
Ehrenberg, D./Krallmann, H./Rieger, B. (Hrsg.): Wissensbasierte Systeme in der Betriebswirtschaft, Erich Schmidt, 1990

[Eicker 91, Metadatenbanksystem], 8-212
Eicker, S.: Anforderungen an ein Metadatenbanksystem. In: HMD 161, 1991, S. 3-9

[Engel 79, Office communications], 2-25; 2-26
Engel, G.H.; Groppuso, J.; Lowenstein, R.A.; Traub, W.G.: An Office Communications System. In: IBM Systems Journal, 3, 18, 1979, S. 402-431

[Engelmore 88, Blackboard system definition], 8-218
Engelmore, R.S./Morgan, A.J./Nii, H.P.: Introduction. In: [Engelmore 88, Blackboard systems], S. 1-24

[Engelmore 88, Blackboard systems], 8-218
Engelmore, R.S./Morgan, A.J.: Blackboard systems, Addison-Wesley, 1988

[Fagan 84, Monitoring Task], 6-143
Fagan, L.; Kunz, J.; Feigenbaum, E.; Osborn, J.: Extensions to the Rule-Based Formalism for a Monitoring Task. In: [Buchanan 84, Rule-Based XPS], Addison-Wesley, 1984, S. Kap. 22

[Fayol 49, Management], 2-21
Fayol, H.: General and industrial Management, Pitman (Ertsveröff. 1916), 1949

[Fick 80, DSS-Issues and challenges], 6-113
Fick, G.P./Sprague, R.-H. (Hrsg.): DSS - Issues and Challenges. Proc. of an International Task Force meeting, June 23-35, 1980, Pergamon-Press, 1980

[GI-Proc. 85, DBS], 6-114
Proc. GI-Fachtagung: Datenbanksysteme für Büro, Technik und Wissenschaft, Springer, 1985

[Gloor 90, Hypermedia], 8-221
Gloor, P.A.: Hypermedia-Anwendungsentwicklung, Teubner, 1990

[Gnesereth 87, Logical Foundations], 6-160
Gnesereth, M.; Nilsson, N.: Logical Foundations of Artificial Intelligence, Morgan Kaufmann, 1987

[Goodwin 82, TMS], 6-160
Goodwin, J.: An Improved Algorithm for Non-Monotonic Dependency Net Update, LITH-MAT-R-82-83, Linköping University, 1982

[Gordon 85, Evidential Reasoning], 6-155
Gordon, J.; Shortliffe, E.: A Method for Managing Evidential Reasoning in a Hierarchical Hypotheses Space. In: AI Journal 26, 1985, S. 323-357

[Greschner 92, Erfolgsfaktor Information], 10-241; 10-243
Greschner, J./Zahn, E.: Strategischer Erfolgsfaktor Information. In: [Krallmann 92, RAP], S. 9-28

[Guida 89, Topics in Expert Systems], 6-140
Guida, G.; Tasso, C. (Hrsg.): Topics in Expert System Design, Methodologies and Tools, North-Holland, 1989

[Gulick 37, Science of Administration], 2-22
Gulick, L.H.; Urwick, L.F. (Hrsg.): Papers on the Science of Administration, Columbia University Press, New York, 1937

[Gulick 37, Theory of Organization], 2-22
Gulick, L.H.: Notes on the Theory of Organization. In: [Gulick 37, Science of Administration]

[Güsgen 89, Constraint Satisfaction], 6-145
Güsgen, H.-W.: CONSAT: A System for Constraint Satisfaction, Pitman, 1989

[Habermann 93, Repository], 8-211; 8-213; 8-215
Habermann, H.-J.; Leymann, F.: Repository - Eine Einführung, Oldenbourg, 1993

[Hales 86, What do managers do?], 2-23
Hales, C.P.: What do managers do? A critical review of the evidence. In: Journal of Management Studies, 23,1, 1986, S. 88-115

[Hampe-Neteler 92, Software-Ergonomie], 10-246
Hampe-Neteler, W.; Rödiger, K.-H.: Software-Ergonomie - Verfahren zur Evaluierung und Standards zur Entwicklung von Benutzeroberflächen, Universität Bremen, FB Informatik, Bericht Nr. 2/92, 1992

[Härder 85, DBS-Architektur], 6-114
Härder, T.; Reuter, A.: Architektur von Datenbanksystemen für Non-Standard-Anwendungen. In: [GI-Proc. 85, DBS]

[Hartson 90, Benutzeroberfläche], 5-104
Hartson, H.R,; Hix, D.: Human-Computer Interface Development: Concepts and Systems for Its Management, ACM Computing Surveys 21, 1989

[Hayes-Roth 83, Building Expert Systems], 6-140
Hayes-Roth, F.; Waterman, D.; Lenat, D. (Hrsg.): Building Expert Systems, Addison-Wesley, 1983

[Heilmann 87, Computerunterstützung], 2-36
Heilmann, H.: Computerunterstützung für das Management - Entwicklung und Überblick. In: Handbuch der modernen Datenverarbeitung (HMD), Heft 138, 1987, S. 3-18

[Hentenryck 89, Constraint Satisfaction], 6-145
Hentenryck, P.: Constraint Satisfaction in Logic Programming, MIT-Press, 1989

[Hewitt 77, Control Structures], 8-216
Hewitt, C.E.: Viewing control structures as patterns of passing messages. In: Artificial Intelligence, Vol. 8, No. 3, 1977, S. 323-364

[Hunt 84, Management Behaviour and Leadership], 2-24
Hunt; Hosking; Schriesheim; Stewart (Hrsg.): Leaders and Managers - Interpersonal Perspectives on Management Behaviour and Leadership, Pergamon Press, New York, 1984

[Johannsen 83, man-machine systems], 10-245
Johannsen, G.; Rijnsdorp, J.E. (eds.): Analysis, design and evaluation of man-machine systems, Pergamon, 1983

[Jones 86, EIS], 2-32
Jones, J.W.; McLeod, R.: The Structure of Executive Information Systems: An Exploratory Analysis. In: Decision Sciences, Vol. 17, 1986, S. 220-249

[Keen 80, DSS-Research-Perspective], 6-113
Keen, P.G.W.: DSS - A Research Perspective. In: [Fick 80, DSS-Issues and challenges], S. 23-44

[Kim 89, Object-oriented Concepts], 8-208
Kim, W.; Lochovsky, F. H. (Hrsg.): Object Oriented Concepts, Databases and Applications, ACM Press, 1989

[Kinnebrock 92, Neuronale Netze], 8-223; 8-224
Kinnebrock, W.: Neuronale Netze - Grundlagen, Anwendungen, Beispiele, Oldenbourg, 1992

[Klemmer 72, Time spent communicating], 2-27
Klemmer, E.T; Snyder, F.W.: Measurement of Time Spent Communicating. In: The Journal of Communication, 22, 1972, S. 142-158

[Kobsa 85, Benutzermodellierung], 6-163
Kobsa, A.: Benutzermodellierung in Dialogsystemen, Berlin u.a., 1985

[Kotter 82, The general managers], 2-24
Kotter, J.P.: The general managers, London: Macmillan, 1982

[Krallmann 87, Executive Support System], 1-14; 6-115
Krallmann, H./Rieger, B.: Vom Decision Support System (DSS) zum Executive Support System (ESS). In: Handbuch der modernen Datenverarbeitung, Heft 138, Forkel, November 1987, S. 28-38

[Krallmann 92, RAP], 8-218
Krallmann, H./Papke, J./Rieger, B. (Hrsg.): Rechnergestützte Werkzeuge für das Management, Erich-Schmidt-Verlag, 1992

[Kramer 89, verteilte Systeme], 8-215
Kramer, J.; Sloman, M.: Verteilte Systeme und Rechnernetze, Karl Hanser Verlag, 1989

[Kreutzer 90, OOP], 8-201; 8-202
Kreutzer, W.: Grundkonzepte und Werkzeugsysteme objektorientierter Systementwicklung - Stand der Forschung und Anwendung. In: Wirtschaftsinformatik, Heft 6, 1990

[Kuhlen 91, Hypertext], 8-219
Kuhlen, R.: Hypertext - ein nicht-lineares Medium zwischen Buch und Wissensbank, Springer, 1991

[Kulikowski 82, CASNET/EXPERT], 6-156
Kulikowski, C.; Weiss, S.: Representation of Expert Knowledge for Consultation: the CASNET and EXPERT Projects. In: [Szolovits 82, AI in Medicine]

[Kurbel 89, Expertensysteme], 6-138; 6-141; 6-151; 6-154; 6-163
Kurbel, K.: Entwicklung und Einsatz von Expertensystemen, Springer, 1989

[Kurke 83, Nature of managerial work], 2-30
Kurke, L.B.; Aldrich, H.E.: Mintzberg was right! - A Replication and Extension of the Nature of Managerial Work. In: Management Science, 29, 1983, S. 975-984

[Lieberman 86, Prototypical Objects], 8-208
Lieberman, H.: Using Prototypical Objects to Implement Shared Behaviour in Object-Oriented Systems. In: Proc. 1st ACM Conference on Object-Oriented Systems, SIGPLAN Notices, Vol. 21, No. 9, 1986, S. 214-223

[Lippold 82, Management], 2-36
Lippold, H.: Management und interaktive Systeme, Frankfurt, 1982

[Ljubojevic 91, objektorientiertes Informationsmodell], 8-213
Ljubojevic, M.: Repositories mit objektorientiertem Informationsmodell. In: HMD 161, 1991, S. 44

[Luthans 84, Measuring leader behaviour], 2-24
Luthans, F.; Lookwood, D.L.: Toward an Observation System for Measuring Leader Behaviour in Natural Settings. In: [Hunt 84, Management Behaviour and Leadership], S. 117-141

[Marcus 88, Knowledge Acquisition], 6-140
Marcus, S. (Hrsg.): Automating Knowledge Acquisition for Expert Systems, Kluwer Academic Publ., 1988

[Martial 92, Verteilte Künstliche Intelligenz], 8-215
Martial, F. von: Einführung in die Verteilte Künstliche Intelligenz. In: KI, 1/92, FBO-Verlag, 1992

[McAllister 80, TMS], 6-160
McAllister, D.: An Outlook on Truth Maintenance Systems, AI-Memo-551, MIT, 1980

[McDermott 88, Problem-Solving Methods], 6-140
McDermott, J.: Preliminary Steps Toward a Taxonomy of Problem-Solving Methods. In: [Marcus 88, Knowledge Acquisition], S. 225-256

[Mertens 91, Planungs- und Kontrollsysteme], 5-95
Mertens, P.; Griese, J.: Integrierte Informationsverarbeitung 2 - Planungs- und Kontrollsysteme in der Industrie, 6. Aufl., Gabler, 1991

[Meyer 90, Objektorientierte Softwareentwicklung], 8-202
Meyer, B.: Objektorientierte Softwareentwicklung, Hanser/Prentice-Hall, 1990

[Miller 82, INTERNIST], 6-155
Miller, R.; Pople, H.; Myers, J.: INTERNIST1, an Experimental Computerbase Diagnostic Consultant for General Internal Medicine. In: New England Journal of Medicine 307, 1982, S. 468-476

[Minsky 75, Frames], 6-142
Minsky, M.: A Framework for Representing Knowledge. In: [Winston 75, Psychology of Computer Vision]

[Mintzberg 73, Managerial Work], 2-21; 2-22; 2-23; 2-29
Mintzberg, H.: The Nature of Managerial Work, Harper&Row, 1973

[Morton 83, MSS], 1-13
Morton, M.S. Scott: State of the Art of Research in Management Support Systems. CISR-MIT, Working Paper #107, 1983

[Müller-Böling 90, IuK-Anforderungen für Führungskräfte], 2-36; 5-101
Müller-Böling, D.: Aufgabenbedingte und persönlichkeitsbedingte Anforderungen an Informations- und Kommunikationstechniken für Führungskräfte. In: [GI 90, 20.Jahrestagung-I], S. 92-111

[Nastansky 90, Endbenutzercomputing], 8-201
Nastansky, L.: Objektorientierte Systeme im Endbenutzercomputing. In: Wirtschaftsinformatik, Heft 3, 1990, S. 238-252

[Nastansky 90, Hypermediabasiertes Informationsmanagement], 8-220; 8-222
Nastansky, L.; Seidensticker, F.-J.: Anwendungen und Konzepte für Hypermedia-basiertes Informationsmanagement am netzintegrierten Managerarbeitsplatz. In: Wirtschaftsinformatik, Heft 6, 1990, S. 519-537

[Nastansky 92, Hypermedia], 8-218; 8-220
Nastansky, L.: Hypermedia - Grundlagen und Anwendungsperspektiven für das Management.. In: [Krallmann 92, RAP], S. 123-142

[Neumann 87, Wissensbasierte Planung], 6-168
Neumann, B.; Cunis, R.; Günter, A.; Syska, I.: Wissensbasierte Planung und Konfigurierung. In: Proc. GI-Kongreß "Wissensbasierte Systeme", Informatik Fachberichte 155, Springer, 1987

[Newell 82, Knowledge Level], 6-140
Newell, A.: The Knowledge Level. In: AI-Journal 18, 1982, S. 86-127

[Nierstranz 89, Object-Oriented Concepts], 8-208
Nierstranz, O.: A Survey of Object-Oriented Concepts. In: [Kim 89, Object-Oriented Concepts], S. 3-21

[Nilsson 82, Artificial Intelligence], 6-141; 6-167
Nilsson, N.: Principles of Artificial Intelligence, Springer, 1982

[Panko 88, Enduser Computing], 2-25
Panko, R.R.: Enduser Computing - Management, Application, Technology, Wiley&Sons, 1988

[Pauliks 91, Marketing-Controlling], 5-108
Pauliks, M.: Konzeption und Implementierung einer computergestützten Planungs- und Kontrollkomponente für das Marketing-Controlling eines Berliner Anlagenbauers, Studienarbeit, Technische Universität Berlin, 1991

[Puppe 83, MED1], 6-156
Puppe, F.: MED1: Ein heuristisches Diagnosesystem mit effizienter Kontrollstruktur, Universität Kaiserslautern, Interner Bericht 71/83, 1983

[Puppe 87, Diagnostisches Problemlösungen], 6-144
Puppe, F.: Diagnostisches Problemlösen mit Expertensystemen, Informatik Fachberichte 148, Springer, 1987

[Puppe 90, Problemlösungsmethoden], 6-140; 6-164; 6-167; 6-168
Puppe, F.: Problemlösungsmethoden in Expertensystemen, Springer, 1990

[Puppe 91, Expertensysteme], 6-138; 6-141; 6-144; 6-148; 6-151; 6-154; 6-157; 6-160; 6-162; 6-163; 6-168
Puppe, F.: Einführung in Expertensysteme, 2. Aufl., Springer, 1991

[Reichardt 92, Versicherungs-Controlling], 7-170
Reichardt, K.: Versicherungs-Controlling, Technische Universität Berlin, 1992

[Reichwald 89, Bürokommunikationstechnik], 2-35; 2-36
Reichwald, R./Stauffert, T.: Bürokommunikationstechnik für Führungskräfte - Gibt es ein Nutzungspotential?. In: Office Management, Heft 4, 1989, S. 6-12

[Reimann 85, DSS-Software], 6-128
Reimann, B.C.; Waren, A.D.: User-Oriented Criteria for the Selection of DSS-Software. In: Communications of the ACM, Vol. 28, No.2, 1985, S. 166-179

[Rieger 85, EUS], 6-113
Rieger, B.: Leistungsfähigkeit von Entscheidungs-Unterstützungs-Systemen (EUS) im Unternehmen - dargestellt am Beispiel der Preisgestaltung von Mehrproduktunternehmen, Dissertation, TU-Berlin, 1985

[Rieger 90, EIS-Generatoren], 5-85
Rieger, B.: Vergleich ausgewählter EIS-Generatoren. In: Wirtschaftsinformatik, Heft 6/90, S. 503-518

[Rieger 90, EIS-State of the art], 8-209
Rieger, B.: Executive Information Systems (EIS): State-of-the-art und Zukunftsperspektiven. (Tutorial). In: [DGOR 91], S. 350-357

[Rieger 90, Wissensbasierte Planungssprachen], 3-58; 4-68; 6-127; 6-136
Rieger, B.: Wissensbasierte Erweiterungen von Planungssprachen. In: [Ehrenberg 90, Wissensbasierte Systeme], S. 251-266

[Rieger 93, EIS-Potentiale], 8-225
Rieger, B.: Entwicklungspotentiale und -tendenzen für Executive Information Systems (EIS). In: DV-Management, Heft 3, 1993

[Ritter 91, Neuronale Netze], 8-224
Ritter, H.; Schulten, K.; Martinetz, T.: Neuronale Netze: eine Einführung in die Neuroinformatik selbstorganisierter Netzwerke, Addison-Wesley, 2. erw. Aufl., 1991

[Roberts 77, FRL], 6-146
Roberts, B.; Goldstein, I.: The FRL-Primer, AI-Memo 408, MIT, 1977

[Rockart 88, Executive Support Systems], 3-49; 3-50; 3-51
Rockart, J.F.; DeLong, D.W.: Executive Support Systems, Dow Jones-Irwin, Homewood Illinois, 1988

[Round 89, KB-Simulation], 6-168
Round, A.: Knowledge-Based Simulation. In: [Barr 89, Artificial Intelligence], Vol. IV, S. 415-518

[Scheer 90, Wirtschaftsinformatik], 5-75; 8-213
Scheer, A.-W.: Wirtschaftsinformatik - Informationssysteme im Industriebetrieb, 3. Aufl., Springer, 1990

[Schöneburg 92, Neuronale Netze], 8-224
Schöneburg, E.: Industrielle Anwendungen Neuronaler Netze, Addison-Wesley, 1992

[Schönpflug 87, Software-Ergonomie], 10-246
Schönpflug, W.; Wittstock, M. (Hrsg.): Software-Ergonomie '87, Teubner, 1987

[Schorr 88, Graphiktypen], 5-103
Schorr, G.: Entwicklung einer Graphikkomponente für das Expertensystemtool PC-HEXE, Diplomarbeit, Nürnberg, 1988

[Schreyögg 91, Manager-Arbeit], 2-29; 2-31
Schreyögg, G./Hübl, G.: Manager und ihre Arbeit, Diskussionsbeitrag Nr. 159 des FB WW, Fernuniversität Hagen, 1991

[Seidensticker 90, Hypermedia], 8-218; 8-222
Seidensticker, F.-J.: Information Management mit Hypermedia-Konzepten, S+W Steuer- und Wirtschaftsverlag, 1990

[Shapiro 87, Artificial Intelligence], 8-215
Shapiro, S.C.; Eckroth, D. (Hrsg.): Encyclopedia of Artificial Intelligence. In: Vol. 2, John Wiley & Sons, New York, 1987

[Shortliffe 75, Inexact Reasoning], 6-156
Shortliffe, E.; Buchanan, B.: A Model of Inexact Reasoning in Medicine. In: Math. Bioscience 23, 1975, S. 351-379

[Smith 80, Contract-Net-Protocol], 8-217
Smith, R.: The Contract Net Protocol - High-Level Communication an Control in a Distributed Problem Solver. In: IEEE Transactions on Computers, C-29(12), 1980, S. 1104-1113

[Sol 82, DSS-Processes and Tools], 6-113
Sol, H.G. (Hrsg.): Processes and Tools for Decision Support. Proc. of the Joint IFIP WG 8.3/IIASA Working Conference, Schloss Laxenburg, Österreich, Juli 1982

[Sprague 80, DSS-Research-Framework], 6-113
Sprague, R.H.: A Framework for Research on Decision Support Systems. In: [Fick 80, DSS-Issues and challenges], S. 5-22

[Stachowiak 73, Modelltheorie], 6-127
Stachowiak, H.: Allgemeine Modelltheorie, Springer, 1973

[Stahlknecht 91, Wirtschaftsinformatik], 8-215
Stahlknecht, P.: Einführung in die Wirtschaftsinformatik, 6. Aufl., Springer, 1993

[Stallman 77, Circuit Analysis], 6-145
Stallman, R.; Sussman, G.: Forward Reasoning and Dependancy Directed Backtracking in a System fpr Computer-Aided Circuit Analysis. In: AI-Journal 9, 1977, S. 135-196

[Steiner 92, Kooperation], 8-217
Steiner, D.; Haugeneder, H.; Kolb, M.; Bomarius, F.; Burt, A.: Mensch-Maschine-Kooperation. In: KI, 1992

[Stewart 67, Managers and their Jobs], 2-23
Stewart, R.: Managers and their Jobs, London: Macmillan, 1967

[Stewart 76, Contrasts in Management], 2-23
Stewart, R.: Contrasts in Management, Maidenhead, McGraw-Hill, 1976

[Stohr 83, DSS-User interfaces], 6-113
Stohr, E.A./White, N.H.: User Interfaces for DSS - An Overview. In: International Journal of Policy and Information Systems, Vol. 6, 1983

[Stülpnagel 91, Repositories], 8-211
Stülpnagel, A. von: Repositories - Konzepte, Architekturen, Standards. In: HMD 161, 1991, S. 10-25

[Sundermeyer 92, verteilte WBS], 8-216; 8-217
Sundermeyer, K.: Kooperierende verteilte wissensbasierte Systeme, Seminarunterlagen SS 92, TU-Berlin, 1992

[Sussman 80, Constraints], 6-145
Sussman, G.; Steele, G.: Constraints - a Language for Expressing Almost Hierarchical Descriptions. In: AI-Journal 14, 1980, S. 1-39

[Szolovits 78, Categorical and Probabilistic Reasoning], 6-157
Szolovits, P.; Pauker, S.: Categorical and Probabilistic Reasoning in Medical Diagnosis. In: AI Journal 11, 1978, S. 115-144

[Szolovits 82, AI in Medicine], 6-156
Szolovits, P. (Hrsg.): Artificial Intelligence in Medicine, AAAS Selected Symposium 51, Westview Press, 1982

[Tiemeyer 89, PC-Nutzung], 2-36
Tiemeyer, E./ Herzog, F.: PC-Nutzung durch Führungskräfte. In: Office Management, Heft 4, 1989, S. 28-37

[Ulich 91, Arbeitspsychologie], 10-245
Ulich, E.: Arbeitspsychologie, Poeschel, 1991

[Vallone 91, computergestützte Arbeitsumgebung], 2-23; 2-25
Vallone, C.: Designkonzepte einer kommunikationsorientierten computergestützten Arbeitsumgebung für das Management, Dissertation Nr. 1254, Hochschule St. Gallen, 1991

[Wagner 80, DSS], 10-241
Wagner, G.R.: Optimizing DSS. In: Datamation, Vol. 26, No. 5, 1980, S. 209-214

[Winston 87, Künstliche Intelligenz], 6-158
Winston, P.H.: Künstliche Intelligenz, Addison-Wesley, 1987

[Witt 92, Objektorientierte Programmierung], 8-208; 8-209
Witt, K.-U.: Einführung in die objektorientierte Programmierung, Oldenbourg, 1992

[Zahn 90, Wettbewerbsfaktor Information], 10-244
Zahn, E.: Informationstechnologie als Wettbewerbsfaktor. In: Wirtschaftsinformatik, 6, 1990, S. 493-502

Stichwortverzeichnis