Inhaltsverzeichnis

1. Einleitung

Kennzahlen sind komprimierte Darstellungen von betrieblichen Sachverhalten. Sie verdichten Informationen und erlauben dadurch eine bessere Kontrolle des Unternehmensgeschehens. Zusammengefaßt in ein Kennzahlensystem, werden sie zu Instrumenten der Führung, bilden eine Grundlage zum Treffen von Entscheidungen.

Betriebswirtschaftliche Kennzahlensysteme dienen vorrangig dem Kosten-Controlling und bewerten die erbrachten Leistungen und die eingesetzten Ressourcen überwiegend monetär. Zur Steuerung der DV-Infrastruktur von Rechenzentren sind neben diesen finanztechnischen und betriebswirtschaftlichen Kennzahlensystemen umfangreiche technische Kennzahlen notwendig. Hierzu zählen die Verfügbarkeit der Netze und Systemplattformen, die durchschnittliche Nutzungsdauer und Nutzungshäufigkeit bestimmter Dienste, die Qualität der Hotline usw.

Die bekanntesten Kennzahlensysteme, die den technischen Aspekt des RZ-Betriebs abdecken, stammen aus den 80'er Jahren. Nennenswert sind die Beiträge der Schweizerischen Vereinigung für Datenverarbeitung ([SVD80]), das Diebold-Kennzahlensystem ([Dieb84]), die Systeme von Braun und Grohmann ([Brau81], [Groh79]). Auch heute werden wissenschaftliche Arbeiten auf diesem Gebiet veröffentlicht (z.B. [Wies96-3]).

Ein wesentliches Merkmal aller bisherigen Ansätze ist die Definition von konkreten Kenngrößen und festen Ordnungsstrukturen für diese Größen. So werden z.B. in [SVD80] 31 Kennzahlen definiert und in eine Pyramide mit den Ebenen Management-, Benutzer- und EDV-Mitarbeiter-Kennzahlen eingeordnet. In dem Diebold-Kennzahlensystem werden 106 Kennzahlen gebildet, rechentechnisch miteinander verknüpft und pyramidenförmig verdichtet.

Wohl keines dieser Systeme könnte in einem konkreten Rechenzentrum ohne Änderungen und Anpassungen verwendet werden. Aus der heutigen Sicht ist es fraglich, ob die Kennzahlen "Anzahl der Platten- und Bandwechsel" oder "Anzahl der Listen und Druckzeilen" eine große Bedeutung für die Steuerung eines modernen Rechenzentrums haben[1]. Außerdem hat jedes Rechenzentrum, abhängig von seiner Zielsetzung, seine eigenen Gewichtungen und kann die Kennzahlen anders einordnen wollen, als daß Standardsystem es tut[2].

In dieser Arbeit wird eine neue Methodik zum Aufbau von technischen Kennzahlensystemen für Rechenzentren vorgestellt. Im Unterschied zu den bisherigen Ansätzen legen wir die Gewichtung auf die Ausarbeitung eines Frameworks, d.h. eines vollständigen und in sich geschlossenen Rahmenkonzeptes zur Definition und Realisierung von *beliebigen* technischen RZ-Kennzahlensystemen. Zum Beweis der Flexibilität des Konzeptes erarbeiten wir drei konkrete Kennzahlen-Subsysteme und zeigen, wie sie realisiert werden können.

Die Evaluierung der Praxistauglichkeit des Frameworks erfolgt anhand der Anforderungen des Leibniz-Rechenzentrum der Bayerischen Akademie der Wissenschaften (kurz: LRZ). Wir ver-

---

[1] Diese zwei Kennzahlen sind Bestandteile der Systeme von [Brau81] und [Groh79].

[2] Die Kennzahl L6 "Durchschnittliche Antwortzeit" aus dem SVD-System wird im Original der EDV-Mitarbeiter-Ebene zugeordnet. Es existieren aber durchaus Situationen, wo die Benutzer der DV-Leistungen diese Kennzahl auch einsehen wollen (z.B. dann, wenn die zentrale DV ihre Leistungen auf der Grundlage dieser und anderen Kennzahlen abrechnet).

wenden dabei die Ergebnisse unserer “Vorgänger” und integrieren ihre Kennzahlen und Subsysteme in das Kennzahlen-Data-Warehouse[3] des LRZ.

Das Framework selbst besteht aus zwei Teilen - einer Methode zur Definition und einer Methode zur Realisierung von Kennzahlensystemen. Die beiden Methoden werden in den Kapiteln 2 und 3 dieser Arbeit ausführlich behandelt. Im Kapitel 4 werden diese Methoden auf die Probleme des Leibniz-Rechenzentrums angewandt und ein Prototyp des Kennzahlensystems entwickelt. Das letzte Kapitel gibt eine Zusammenfassung der Ergebnisse und einen Ausblick auf die Weiterentwicklung des Prototyps.

[3] Zum Begriff “Data Warehouse” siehe weiter unten.

2. Eine Definitions-Methode für technische RZ-Kennzahlensysteme

In diesem Kapitel wird eine Methode zur Definition von technischen Kennzahlensystemen für Rechenzentren vorgestellt. Es wird kein für alle Rechenzentren gültiges Kennzahlensystem definiert, sondern es wird der Vorgang der Entwicklung eines Kennzahlensystems beschrieben. Die Methode liefert eine Reihe von Lösungsansätzen zur Umsetzung von häufig auftretenden Anforderungen an technische Kennzahlensysteme.

Zuerst werden die Grundlagen von Kennzahlensystemen erklärt (Abschnitt 2.1). Es erfolgt die Definition des Begriffs "Technische RZ-Kennzahl" , "Kennzahlensystem" und "Kennzahlenstruktur". Neue Ideen, wie die Funktionssichtweise auf Kennzahlen und der Kennzahlengraph, werden hier vorgestellt. Wir beschreiben die Klassifizierung von Kennzahlen und Kennzahlensystemen und zeigen, wie Kennzahlensysteme strukturiert werden können. Schließlich ordnen wir die Kennzahlensysteme in betriebliche Führungsaufgaben ein. Die Abschnitte 2.2, 2.3 und 2.4 stellen drei konkrete Kennzahlenstrukturen vor, dabei ist die Substruktur "Benutzergruppen"(Abschnitt 2.4) an ein Teil des SVD-Konzeptes angelehnt, die Idee für die Substruktur "Ziel-Mittel-Hierarchie des Rechenzentrums" (Abschnitt 2.2) kommt aus [Hein76], eine große Zahl von Kennzahlen für die Substruktur "Technische Aspekte des Rechenzentrums" (2.3) wurde aus [Wies96] übernommen.

2.1 Grundlagen von Kennzahlensystemen

2.1.1 Begriffsbildung

In der Arbeit von Mayer ([Maye94]), dem Standardwerk für betriebswirtschaftliche Kennzahlensysteme, werden die betriebswirtschaftliche Kennzahlen als "Zahlen, die Informationen über betriebswirtschaftliche Tatbestände beinhalten" definiert. Die in dieser Arbeit gegebene Definition der technischen RZ-Kennzahlen entspricht größtenteils der Mayer'schen Kennzahlenbeschreibung, unterscheidet sich jedoch in einem Aspekt grundlegend :

Technische RZ-Kennzahlen sind zahlenwertige Funktionen, die Informationen über technische Tatbestände des Rechenzentrums liefern.

Wie man leicht merkt, wollen wir unter Kennzahlen keine Zahlen sondern Funktionen verstehen, in anderen Punkten entspricht unsere Beschreibung weitgehend dem Sinn der Standarddefinition. Im folgenden gehen wir auf die vier Merkmale der technischen RZ-Kennzahlen in der Reihenfolge ihrer Bedeutung ein.

1) Technische Tatbestände eines Rechenzentrums

Die Klärung des Begriffs “Technische Tatbestände eines RZ’s” ist sehr wichtig für die Festlegung des Rahmens dieser Arbeit. Alles, was nicht zu diesen Tatbeständen gehört, soll nicht der Gegenstand der Kennzahlenbildung sein.

In seiner Definition beginnt Mayer die Abgrenzung der betriebswirtschaftlichen Tatbestände mit einer Erklärung des Begriffs “Betrieb”. Er schreibt: “Unter Betrieb ist eine planvoll organisierte Wirtschaftseinheit zu verstehen, in der eine Kombination von Produktionsfaktoren (dispositive und ausführende Arbeit, Betriebsmittel und Werkstoffe) mit dem Ziel erfolgt, Sachgüter zu produzieren und Dienstleistungen bereitzustellen” ([Meye94],S. 1). Eine weitere Konkretisierung folgt: “Der Prozeß der Leistungserstellung und der Leistungsverwertung i.w.S. und alle in diesem Zusammenhang auftretenden Probleme bilden den sachlichen Bereich betriebswirtschaftlicher Kennzahlen und Kennzahlen-Systeme”. In diesem Prozeß will er jedoch nicht alle Aspekte behandeln - die rein juristischen, organisatorischen, personellen und technischen Aspekte werden von der Einbeziehung in die betriebswirtschaftlichen Kennzahlensysteme ausgenommen.

Bei der Definition der technischen Tatbestände oder Aspekte eines Rechenzentrums ist es hilfreich die Definition des Betriebs (ein Rechenzentrum ist ohne Zweifel ein Betrieb) etwas gründlicher anzusehen. Ein Rechenzentrum hat mehrere Arbeitsprozesse (Workflows), in denen das Personal, die RZ-Technik, die Werkstoffe und die Produkte und Dienste von externen Herstellern zu seinen eigenen Produkten und Diensten kombiniert werden. Nach dieser Klarstellung ist die Abgrenzung der technischen Aspekte der Betriebsart “Rechenzentrum” problemlos möglich. Der Einsatz der RZ-Technik während der Leistungserstellung und -verwertung und die dabei auftretenden Probleme bilden die technischen Tatbestände von Rechenzentren.

Nicht nur der eigentliche Betrieb der RZ-Technik ist zu den technischen Tatbeständen zuzuordnen. Die Technik wird bei der Automatisierung der Workflows eingesetzt, außerdem gehört die Bereitstellung der technischen Infrastruktur zu den Dienstleistungen des Rechenzentrums. Wir konkretisieren den Begriff “technische Tatbestände” deshalb folgendermaßen:

Unter technischen Tatbeständen (Aspekten) eines Rechenzentrums werden die technischen Betriebsmittel, die Workflows und die Dienstleistungen/Produkte und ihre technischen Charakteristika (also keine finanzwirtschaftliche Charakteristika wie z.B. Kosten) verstanden.

Die technischen Tatbestände sind mehr betriebsspezifisch, als betriebswirtschaftliche Tatbestände. Es gibt allgemeine betriebswirtschaftliche Kennzahlensysteme (DuPont-System, ZVEI-System, andere), jedoch keine für alle Betriebe gleich gut geeignete technische Kennzahlensysteme. Die Begründung dafür ist, daß die betriebswirtschaftlichen Aspekte langlebig und allgemein sind, sie werden nicht jede 8-10 Jahre obsolet, wie es häufig mit der Technik, besonders in Rechenzentren, ist. Die Kennzahlensysteme, die technische Aspekte behandeln, müssen deshalb an die konkreten Betriebe angepaßt werden, durch wissenschaftliche Arbeiten lassen sich bestenfalls Rahmensysteme bzw. Frameworks entwickeln. Deshalb liegt die Gewichtung dieser Arbeit auf der Ausarbeitung solch eines Frameworks zur Definition und Realisierung von technischen RZ-Kennzahlensystemen.

2) <u>Informationen</u>

Informationen sind zweckorientiertes Wissen ([Meye94]). Die Zweck- und Verwendungsorientierung des Wissens im Hinblick auf bestimmte Zielsetzungen und Aufgaben bildet den wesentlichen Bestandteil der Information und unterscheidet das Wissen von der Information. Dazu ein Beispiel:

Wissen: Die Workstation A fiel am So., den 1 Dez. 1997 um 12:00 wegen eines Betriebssystem-Fehlers aus. **Information**: Die Workstations mit dem Betriebssystem X fallen zwei mal häufiger wegen BS-Fehler aus, als Workstations mit anderen Betriebssystemen.

Die Zweckorientierung ist in dem obigen Beispiel der Vergleich von verschiedenen Betriebssystemen in Bezug auf Verläßlichkeit. Sie erfordert u.a. die Verdichtung des Wissens, denn nur so können untypische Wissenseinheiten ausgefiltert werden (so ist z.B. eine Situation vorstellbar, wo eine bestimmte Workstation mit dem Betriebssystem X eine Zeit lang die höchste Verfügbarkeit hat und so die Tendenz verfälscht). Die Informationsaussagen basieren deshalb i.d.R. auf mehreren Wisseneinheiten.

3) Funktionen.

Die Definition von Kennzahlen als Funktionen ist neu. Die meisten Autoren, die sich mit den theoretischen Grundlagen von Kennzahlensystemen befassen, definieren die Kennzahlen als Zahlen. Diese Sichtweise ist jedoch teilweise widersprüchlich. So haben die Kennzahlen verschiedene Werte z.B. in Abhängigkeit von der Zeit. Außerdem sind viele Kennzahlen miteinander durch Rechenausdrucke verknüpft. Das sind alles Charakteristika von Funktionen und nicht von Zahlen.

Wir würden den Funktionsaspekt nicht so stark hervorheben, wenn dieser mathematische Formalismus nicht so große Vorteile bei der Realisierung und Handhabung der Kennzahlensysteme anböte. So bieten die Kennzahlenfunktionen eine theoretische Grundlage für solche Fragestellungen wie “Kennzahlenabhängigkeiten”. Die sehr wichtige Frage nach der Veränderung einer Kennzahl bei Veränderung einer anderen Kennzahl kann auf die Untersuchung von

$$\frac{\partial K_1}{\partial K_2} \cdot \frac{K_2}{K_1}$$

(der sog. Elastizität) abstrahiert werden.

Zusätzlich zu den klaren theoretischen Grundlagen bietet die Funktionssichtweise eine standardisierte und einfache Realisierungsmöglichkeit der Kennzahlen in einem realen Kennzahlensystem. Mathematisch gesehen, ist eine Funktion eine besondere Art von Relationen, d.h. eine Funktion ist formal eine Menge von Tupeln. Somit kann eine Kennzahl auf eine Tabelle oder eine Sicht in einer relationalen Datenbank abgebildet werden. Diese Vorgehensweise ist auf jede beliebige Kennzahl anwendbar, sie bildet deshalb einen Teil unseres Frameworks zum Aufbau von Kennzahlensystemen.

Ein weiterer Vorteil der Funktionssichtweise ist die Reduktion der Anzahl von Kennzahlen, die in einem realen System definiert werden müssen. Durch die Integration von zusätzlichen Parametern in die Kennzahlenfunktion lassen sich allgemeine Kennzahlen bilden, welche die Implementierungsaspekte vollständig abstrahieren. So kann z.B. eine allgemeine Kennzahl “Verfügbarkeit” definiert werden, die sowohl für Anwendungen, als auch für Plattformen und Netze über die gleiche Schnittstelle abfragbar ist. Für den Anwender des Kennzahlensystems stellt das eine Verbesserung des Benutzerinterfaces dar[4].

[4] Leider wird die Implementierung dadurch etwas schwieriger.

Im Abschnitt 2.1.2 gehen wir auf den Funktionsaspekt etwas näher ein.

4) <u>Zahlenwerte</u>
In allen dem Autor dieser Arbeit verfügbaren Quellen werden unter Kennzahlen nur quantifizierbare (in Zahlen und Meßeinheiten darstellbare) Größen verstanden. In dieser Arbeit wird diese Auffassung beibehalten. Die Gründe dafür sind die folgenden:

a) Quantifizierbare Größen lassen sich vergleichsweise leicht mit einem Computer berechnen, was man von qualitativen Größen nicht behaupten kann. Letzteres ist nur mit Expertensystemen auf der Basis von Künstlicher Intelligenz, speziell Fuzzy Logic, realisierbar.
b) Die Quantifizierung erlaubt eine rechentechnische Verknüpfung der Kennzahlen (z.B. Definition der Kennzahl K4 als $K4 = K2 \times K1 + K3$). Dies ist eine nützliche Eigenschaft, die sehr intensiv in betriebswirtschaftlichen Kennzahlensystemen genutzt wird(→ Stichwort "Spitzenkennzahl").
c) Die Beschreibungsfähigkeit der Kennzahlen leidet darunter kaum, da nur wenige Sachverhalte sich nicht quantifizieren lassen (selbst qualitative Merkmale wie "ausreichend", "gut", "sehr gut" lassen sich durch Zuordnung von "Gewichten" in Zahlen bemessen).

☑

Nach der Klärung des Kennzahlenbegriffs kommen wir jetzt zu den Kennzahlensystemen. Der Begriff "Kennzahlensystem" ist nur teilweise standardisiert (vgl. Diskussion in [März83], S. 65 ff.). In dieser Arbeit wird darunter folgendes verstanden:

Technisches Kennzahlensystem eines Rechenzentrums ist eine Zusammenstellung von mehreren Kennzahlen, die technische Aspekte von Rechenzentren beschreiben und nach beliebigen Strukturierungsprinzipien eingeordnet sind.

Um zwischen der Obermenge "Kennzahlensystem" und der Verbindungsstruktur der Kennzahlen in diesem System inhaltlich unterscheiden zu können, wird in dieser Arbeit der Begriff "Kennzahlenstruktur" verwendet (siehe Abbildung 2-1 Einordnung der Begriffe Kennzahlenmenge, -system und -struktur). Für uns besteht ein Kennzahlensystem aus einer Menge von Kennzahlen und einer Kennzahlenstruktur. Die Kennzahlenstruktur bzw. ihre Substrukturen werden dazu verwendet, um Zusammenhänge der Kennzahlen zu verdeutlichen und die Navigation des Benutzers im System zu erleichtern.

Häufig besteht der Nutzen eines Kennzahlensystems nicht nur in der Bereitstellung von Kennzahlen, sondern auch in der Visualisierung der Zusammenhänge zwischen den Kennzahlen und somit auch zwischen den technischen Tatbeständen. Ohne Zweifel ist die Feststellung, daß eine Kennzahl mit einer anderen Kennzahl korreliert, eine wichtige Information für die Betreiber des Rechenzentrums.

Ab einer gewissen Größenordnung der Kennzahlenmenge muß die Navigation in dieser Menge durch zusätzliche Maßnahmen erleichtert werden. Besonders in einem Kennzahlensystem, das für die Unterstützung der operativen Steuerung eingesetzt wird, kann die Anzahl der Kennzahlen mehrere Hundert betragen. In solchen Systemen sind sehr detaillierte und spezialisierte Kennzahlen notwendig, die sich nicht in einer Kennzahl vereinigen lassen. Der Benutzer muß deshalb die ihm wichtigen Kennzahlen leicht finden können, ohne dabei alle Namen der Kennzahlen merken zu müssen. Auch hier werden die Kennzahlenstrukturen eingesetzt, welche die Kennzahlen nach besonderen Merkmalen systematisieren und somit eine gewisse Ordnung schaffen.

Neu in unserer Definition der Kennzahlensysteme ist die Zulassung von beliebigen Strukturierungs- und damit auch der Zusammenhangsprinzipien. In vielen Arbeiten, die sich mit der Kennzahlen-Problematik beschäftigen, werden nur zwei Typen von Zusammenhängen zugelassen - die rechentechnischen und die Ordnungs-Zusammenhänge[5]. Es ist fraglich, ob eine derartige Beschränkung sinnvoll und akzeptabel ist.

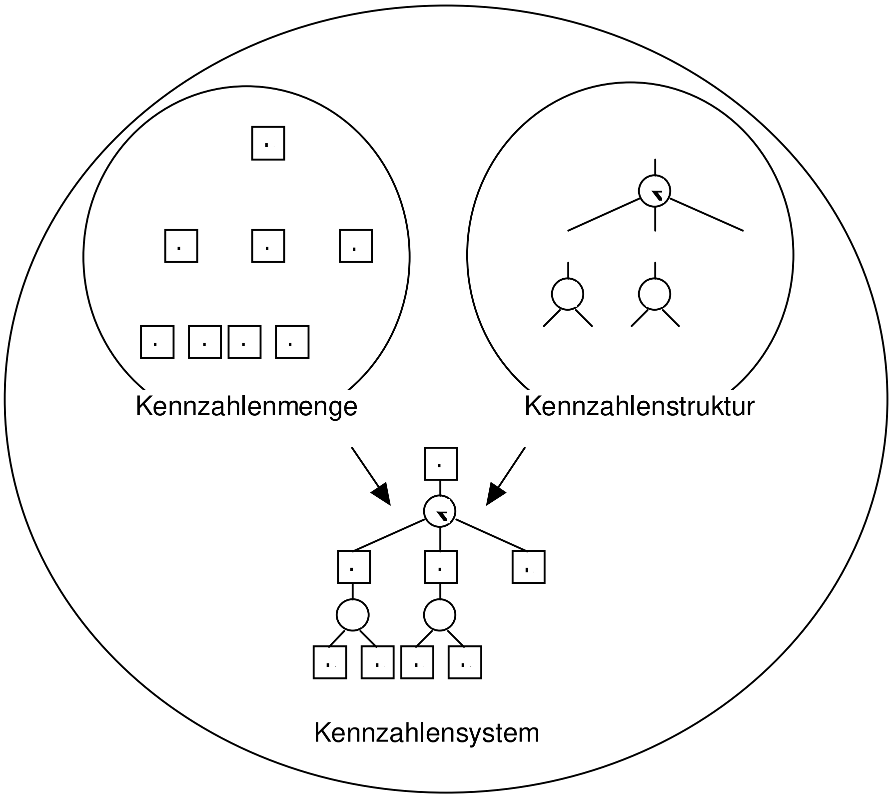


Abbildung 2-1 Einordnung der Begriffe Kennzahlenmenge, -system und -struktur

Um den Entwicklern von Kennzahlensystemen (die Autoren dieser Arbeit inbegriffen) genügend Spielraum bei der Darstellung von Zusammenhängen zu lassen, erlaubt unser Framework beliebige Zusammenhangstypen. Wir nehmen davon selbst Gebrauch und definieren zusätzlich zu den zwei in der einschlägigen Literatur üblichen Strukturierungstypen noch zwei dazu (mehr darüber im Abschnitt 2.1.5).

Das Framework unterstützt die Vielfalt der darstellbaren Zusammenhänge dadurch, daß ein Kennzahlensystem als ein Graph modelliert wird. In der Abbildung 2-1 sieht man eine graphenartige Struktur, die eine Menge von Kennzahlen zu einem Kennzahlensystem verknüpft.

Aus Realisierungsgründen beinhaltet der Graph nicht nur die Kennzahlenknoten (in der Abbildung werden sie durch Quadrate dargestellt), wir benutzen zusätzlich eine weitere Sorte von Knoten (dargestellt durch Kreise), um die Zusammenhänge darzustellen. Auch die Kanten des Graphen dienen der Verdeutlichung der Zusammenhänge. Die Zusammenhangsknoten bilden gemeinsam mit den Kanten die Kennzahlenstruktur.

Ein Vorteil des Kennzahlengraphen wurde bereits erwähnt (Vielfalt der darstellbaren Zusammenhänge). Ein anderer Vorteil ist bei der Realisierung angesiedelt. Ein Graph läßt sich leicht

[5] Wir geben die Definitionen der verschiedenen Zusammenhangstypen weiter unten im Abschnitt 2.1.5.

implementieren[6], somit kann die Kennzahlenstruktur nicht nur auf dem Papier abgelegt werden, sondern läßt sich auch in einer persistenten Datenstruktur speichern. Dadurch wird es möglich, die Unterstützung der Benutzer bei der Navigation im System mit Hilfe von Software zu vereinfachen.

2.1.2 Kennzahlen als Funktionen mit Zahlenwerten

In der Mathematik und der Informatik unterscheidet man zwischen der sog. Funktionalität und der Berechnungsvorschrift einer Funktion. Üblich sich zwei Schreibweisen für die Funktionalität:

$K: P_1 \times P_2 \times \ldots \times P_n \rightarrow W$ und

$\text{fct } K = (P_1, P_2, \ldots, P_n)W$ [7]

In den obigen Ausdrücken sind P_i die Parametermengen und W die Wertemenge. Als Beispiel für Parametermengen kann man die Zeit und Managementobjekte (Workstations, Router, Hosts) nennen.

Die Funktionalität ist die "Schnittstelle" der Funktion, sie beinhaltet die Typen der Parameter und die Wertemenge. Wenn die Berechnungsvorschrift einer Funktion z.B. aus Performance-Gründen geändert werden muß, so sind die auf dieser Funktion aufbauenden anderen Funktionen davon nicht betroffen, solange die Funktionalität die gleiche bleibt.

Die Unterscheidung zwischen der Funktionalität und der Berechnungsvorschrift ist für die Realisierung von Kennzahlen sehr wichtig. Die Funktionalität ist eine Basis für die Definition einer Datenstruktur, welche die Werte der Kennzahlen beinhalten soll (z.B. Tabelle oder Sicht); die mathematische Berechnungsvorschrift ist die Grundlage für den ausführbaren Code zur Kennzahlenberechnung. Die Funktionalität hat auch die Rolle der Benutzerschnittstelle - der Benutzer braucht nichts von der Berechnungsvorschrift zu wissen, er gibt nur die Parameter der Funktion an und bekommt die Werte der Kennzahl zurück. Beispiel:

[6] Wir verwenden z.B. die mathematische Bibliothek LEDA des MPI für Informatik, Saarbrücken, die die vollständigen Implementierungen von Graphen jeden Typs beinhaltet.

[7] vgl. [Broy92], S. 47

Die Kennzahl K “Plattformverfügbarkeit” könnte als eine Funktion mit der Funktionalität :

$$\textit{Plattformverfügbarkeit: Plattform} \times \textit{Zeit} \rightarrow IR_{[0;100]}$$

definiert werden, wobei *Plattform* die Menge der Systemplattformen, *Zeit* die Zeitdimension und $IR_{[0;100]}$ (reelle Zahlen von 0 bis 100) der Wertebereich sind. Um diese Kennzahl zu implementieren, könnten wir eine Tabelle oder eine Sicht in einer relationalen Datenbank definieren (wir bezeichnen sie als “Wertetabelle” bzw. “Wertesicht” der Kennzahl Plattformverfügbarkeit):

```
CREATE TABLE WERT_VERFUEGBARKEIT(
    PLATTFORM           CHAR (100),
    ZEIT                DATE,
    WERT                NUMBER);
```

bzw.

```
CREATE VIEW WERT_ VERFUEGBARKEIT(
    PLATTFORM           CHAR (100),
    ZEIT                DATE,
    WERT                NUMBER)
SELECT AS …;
```

die alle Werte der Kennzahl beinhaltet. Ein Benutzer, der eine Wochenstatistik über die Verfügbarkeit der Workstation “sun1” haben will, definiert z.B. mit Hilfe eines Standard-Reporting-Tools eine einfache Abfrage:

```
    SELECT  ZEIT,WERT
    FROM    WERT_K
    WHERE   PLATTFORM = ‘sun1’
    AND     ZEIT BETWEEN ‘01-JAN-98’ AND ‘8-JAN-98’
```

und erhält die Ergebnisse entweder in der Form einer Graphik oder als Text zurück. Alternativ kann ein Satz von vorgefertigten Abfragen zur Verfügung gestellt werden, so daß der Benutzer nur die passende Abfrage auszuwählen braucht. Natürlich müssen die Entwickler des Kennzahlensystems ein System bereitstellen, das die Wertetabellen der Kennzahlen regelmäßig aktualisiert. Wir zeigen in dem Kapitel 3, wie so ein Aktualisierungssystem entwickelt werden kann.

Die Funktionsvorschriften von Kennzahlen können sich entweder auf andere Kennzahlen abstützen (nichtterminale Kennzahlen) oder terminal sein, d.h. die Werte direkt aus den Basisdaten berechnen. Beispiele für nichtterminale Funktionsvorschriften sind:

- $+[K_1,K_2]$ - Summe von Kennzahlen(-funktionen), definiert als
$$+[K_1,K_2](p_1,\ldots,p_n) = K_1(p_1,\ldots,p_n) + K_2(p_1,\ldots,p_n)$$

- $\times[K_1,K_2]$ - Produkt von Kennzahlen(-funktionen), definiert als
$$\times[K_1,K_2](p_1,\ldots,p_n) = K_1(p_1,\ldots,p_n) \times K_2(p_1,\ldots,p_n)$$

- $1/n\sum[\Delta t,n,K_1]$ - Durchschnittswert einer Kennzahl(-enfunktion) in den letzten Δt Zeiteinheiten bei Abtastfrequenz $\Delta t/n$, definiert als
$$1/n\sum[\Delta t,n,K](t) = \frac{1}{n}\sum_{i=1}^{n} K\left(t - i\frac{\Delta t}{n}\right)$$

Als Beispiele für terminale Kennzahlenfunktionen können folgende Vorschriften genannt werden:

- ipInReceives - Anzahl von In-Paketen eines IP-Ports, definiert als

  ipInReceives(ip-device,t) = Wert von *iso.org.dod.internet.mgmt.mib-2.ip.ipInReceives*[8] des ip-device zum Zeitpunkt t.

- #TroubleTickets - Anzahl der im Zeitintervall (T-anfang, T-ende) eingegangenen und abgeschlossenen Trouble-Tickets, definiert als

  #TroubleTickets(T-anfang, T-ende) =
  $$|\{ TT \in TROUBLETIC \mid TT.Eingang \geq T - anfang \wedge TT.Abschluß < T - ende\}|$$

  falls die Daten in einer Trouble-Ticket-Datenbank mit einer Tabelle “TROUBLETIC” abgespeichert sind.

2.1.3 Klassifizierung der Kennzahlen

Es gibt viele Merkmale, anhand deren man die Kennzahlen klassifizieren kann. Für der Benutzer eines Kennzahlensystem ist die Systematisierung der Kennzahlen anhand dieser Merkmale jedoch von einer geringen Bedeutung. März schreibt dazu in seiner Arbeit ([März83] S. 23), daß “...bei der Unternehmensanalyse auf der Basis von Kennzahlen der Klassifikation der Kennzahlen kein großes Gewicht beigemessen werden sollte. Soll nämlich mit einer Kennzahl eine bestimmte Fragestellung beantwortet werden, dann ist es für den Anwender weitgehend unerheblich, um welchen Kennzahlentyp es sich im einzelnen Fall handelt. Die Einteilung von Kennzahlen hat (...) ausschließlich bei der Konzeption von Kennzahlensystemen Bedeutung, wo eine gewisse Systematisierung erforderlich ist, um dem Anwender des Kennzahlensystems den Umgang damit zu erleichtern ...”. Die im folgenden vorgestellten Merkmale werden wir deshalb hauptsächlich zur Entwicklung der Kennzahlen und Kennzahlenstrukturen verwenden.

1) <u>Managementbereiche.</u>

[8] Entsprechende MIB-Variable des IP-Ports.

In einem technischen RZ-Kennzahlensystem lassen sich folgende Managementbereiche identifizieren: Netze, Systemplattformen, Anwendungen, Gesamtbetrieb. Der letzte Bereich ist die Obermenge der drei ersten.

2) Mathematisch-statistische[9] Gesichtspunkte

Bei diesem Kriterium werden auf der Basis des Typs der Funktionsvorschrift der Kennzahl folgende Gruppen gebildet:

- Absolute Kennzahlen (Kennzahlen mit Kennzahlenfunktion $+(x,y)$, $-(x,y)$, $1/n\sum[\Delta t,n,K](t)$)
- Verhältniskennzahlen (Kennzahlen mit Kennzahlenfunktion $\div(x,y)$)
- Andere Kennzahlen (Kennzahlen mit Kennzahlenfunktion $\times(x,y)$ usw.)

3) Zeitliche Struktur.

Die Zeit ist eine besondere Dimension, die praktisch in jede Kennzahl(-enfunktion) integriert ist. Unterschiede gibt es nur im Bezug auf Zeitpunkte oder Zeiträume. So ist z.B. die Kennzahl ipInReceives, die wir im vorherigen Abschnitt vorgestellt haben, eine zeitpunktbezogene Kennzahl, die Kennzahl #TroubleTickets bezieht sich dagegen auf Zeiträume.

4) Planungsgesichtspunkte.

Viele Autoren teilen die Kennzahlen in Soll- und Ist-Kennzahlen. Mit den Sollzahlen will man bestimmte Werte vorgeben und nach einer bestimmten Zeit mit den tatsächlich erzielten Werten (Ist-Kennzahlen) vergleichen. Zu jedem Zeitpunkt existieren somit Soll- und Ist-Werte, deren Abweichung entweder positiv (z.B. größere Verfügbarkeit als geplant) oder negativ (z.B. größerer Papierverbrauch als angenommen) zu bewerten ist und entsprechende Steuerungsmaßnahmen hervorrufen muß. Obwohl die Idee der Führung von zwei parallelen Werten sehr nützlich für die Unterstützung der Steuerung ist, müssen in üblichen Kennzahlensystemen für jedes Tatbestand zwei Kennzahlen definiert werden (die Soll- und die Ist-Kennzahl). Wir zeigen weiter unten, wie diese künstliche Teilung durch Funktionssichtweise auf Kennzahlen aufgehoben werden kann.

5) Standardisierung.

Viele Rechenzentren haben gleichlaufende Arbeitsprozesse, setzen ähnliche Technik ein und bieten standardisierte Dienste an. Für häufig vorkommende Tatbestände lassen sich standardisierte Kennzahlen einsetzen - das erleichtert die Entwicklung eines Kennzahlensystems erheblich. Als Beispiel für solche Standardkennzahlen läßt sich die “Verfügbarkeit” nennen, die als Verhältnis der Ist-Betriebszeiten zu Soll-Betriebszeiten definiert wird ([SVD80]). Ein weiterer Vorteil beim Einsatz von Standards ist die Vergleichbarkeit der Rechenzentrumsleistung. So können z.B. direkte Schlußfolgerungen gemacht werden, wenn das Rechenzentrum A eine 95%-Verfügbarkeit und das Rechenzentrum B die Verfügbarkeit von 99,5% hat.

Trotz vieler Gemeinsamkeiten bestehen zum Teil gravierende Unterschiede, z.B. zwischen den Profit Centern und den internen DV-Abteilungen. Ein Teil der Kennzahlen in einem Kennzahlensystem wird immer betriebsindividuell bleiben.

6) Benutzer der Kennzahlen.

[9] In [Maye94] und anderen Quellen auch als “statistisch-methodische” Gesichtspunkte bezeichnet

Als Benutzergruppen lassen sich meistens die Anwender, die Mitarbeiter des RZ und das Management des RZ bestimmen. Diese Gruppen haben einen unterschiedlichen Bedarf an Kennzahlen, der sich jedoch überlappen kann.

7) <u>Leistung / Nutzen / Kosten.</u>

Die Autoren des SVD-Systems unterscheiden zwischen Leistungs-, Nutzen und Kostenkennzahlen. An sich genommen bietet diese Unterteilung gute Anhaltspunkte bei der Navigation und dem Umgang mit dem Kennzahlensystem. Wir denken, daß ein allgemeines RZ-Kennzahlensystem in irgendeiner Form diese Unterteilung beinhalten sollte. Dieses Unterscheidungsmerkmal scheitert aber in unserem Fall an dem Umfang der Arbeit (technische Kennzahlen, also keine Kosten; schwierige Ermittlung des "Nutzens"). Alle bei uns vorgestellten Kennzahlen sind also aus dem Bereich "Leistung".

☑

Die sieben Merkmale der Kennzahlenklassifizierung sind für die Benutzer des Kennzahlensystems von einer geringen Bedeutung, für die Entwickler der Kennzahlensysteme erweisen sie sich teilweise als nützlich.

Die Klassifizierungskriterien "Managementbereiche" und "Kennzahlen-Benutzer" werden in dieser Arbeit dadurch berücksichtigt, daß wir eigene Kennzahlenstrukturen von ihnen ableiten. Diese Strukturen zeichnen sich dadurch aus, daß sie alle[10] Kennzahlen des Kennzahlensystems beinhalten (eine Kennzahl wird zumindest von einer Benutzergruppe empfangen, sonst hat sie keine Existenzberechtigung; eine Kennzahl liefert Informationen über mindestens einen der vier Managementbereiche). Somit können sie als sog. Filter eingesetzt werden. Mehr über diese Einsatzmöglichkeit steht im Abschnitt 2.4.

Wie bereits erwähnt, werden die Planungsgesichtspunkte und die zeitliche Struktur durch die Funktionssichtweise auf Kennzahlen vollständig abgedeckt. Die Soll- und Ist-Kennzahlen lassen sich als Funktionswerte einer allgemeinen Kennzahl integrieren. Die Funktionalität der Kennzahlenfunktion ist dafür um die Dimensionen Zeit und "Planungssicht" zu erweitern:

$$K: P_1 \times \ldots \times P_n \times \text{Zeit} \times \{\text{Soll},\text{Ist}\} \rightarrow W$$

Beispiel:

[10] Mit der Ausnahme von Hilfskennzahlen, die nur zur aus rechentechnischen Gründen eingeführt werden, damit die Berechnung von anderen Kennzahlen einfacher ist.

Die Vorgabe

"Die Workstation sun1 soll am 31.12.1997 volle 24 Stunden im Betrieb bleiben"

wird als Tupel

Betriebszeiten(sun1, 31.12.1997, Soll , 24)

in die Kennzahl “Betriebszeiten” mit der Funktionalität:

$$Betriebszeiten: Plattform \times Zeit \times \{Soll, Ist\} \rightarrow IR$$

integriert. Falls die Workstation dann doch für eine Stunde runtergefahren werden muß und die tatsächliche Einschaltzeit nur 23 Std. beträgt, so ist ein neuer Tupel in der Funktion “Betriebszeiten” zu erzeugen:

Betriebszeiten(sun1, 31.12.1997, Ist , 23)

Die Kennzahl “Plattformverfügbarkeit”, die als

$$Plattformverfügbarkeit(p,t) := \frac{Betriebszeiten(p,t,Ist)}{Betriebszeiten(p,t,Soll)}$$

definiert ist, wird am 1.1.98 um einen neuen Wert erweitert:

Plattformverfügbarkeit(sun1, 31.12.1997, 95.8[11]).

2.1.4 Klassifizierung der Kennzahlenstrukturen

Im Unterschied zu der Kennzahlenklassifizierung ist die Klassifizierung der Kennzahlenstrukturen wichtig sowohl für die System-Entwickler als auch für die Anwender. Da die Strukturen abhängig von ihrem Typ unterschiedliche Zusammenhänge darstellen und zur Unterstützung von verschiedenen Kontrollaufgaben konzipiert sind, muß der Anwender zumindest mit der Grundlagen der Klassifizierung vertraut sein, um das Kennzahlensystem effektiv benutzen zu können. Die Klassifizierungsmerkmale und die verschiedenen Strukturtypen werden in eigenen Abschnitten (2.1.5 und 2.1.6) ausführlich diskutiert. In diesem Abschnitt werden sie nur kurz vorgestellt.

Es gibt folgende wichtige Klassifizierungsmerkmale für Kennzahlenstrukturen:

1) Strukturierungsart der Elemente

Wir erwähnten bereits im Abschnitt 2.1.1, daß unser Framework beliebige Kennzahlenstrukturen unterstützt. In der Literatur findet man zur Zeit aber nur vier Arten von interessanten Kennzahlen-Zusammenhängen: rechentechnische Verknüpfungen, Ordnungs-Zusammenhänge, Heuristiken und empirisch-statistische Kausalitäten[12]. Entsprechend unterscheidet man zwischen

- Rechenstrukturen,

[11] Der Wert “95.8 %” kommt aus dem Bruch “23/24”.
[12] Die letzten zwei Zusammenhänge sind nur in [März83] zu finden.

- Ordnungsstrukturen,
- Heuristischen Strukturen und den
- Empirisch-statistischen Strukturen.

2) Verwendungsorientierung

Die Kennzahlensysteme sind ein Teil der betrieblichen Führungsinstrumente. Der Prozess der Führung bzw. Ausführung wird in Planung, Steuerung, Kontrolle, Analyse und den zwischenbetrieblicher Vergleich (best case practice) geteilt (vgl. [Meye94]). Die Unterstützung dieser Aufgaben erfolgt bei uns über die einzelnen Substrukturen des Gesamtsystems, deshalb unterscheiden wir zwischen

- Planungs-,
- Steuerungs-,
- Kontroll-,
- Analyse-Kennzahlenstrukturen und den
- Strukturen für den zwischenbetrieblichen Vergleich.

☑

Eine wichtige Bemerkung vorweg. Im Kapitel 2.1.1 "Begriffsbildung" wurde darauf hingewiesen, daß wir in dieser Arbeit zwischen den Begriffen Kennzahlenstruktur und -system unterscheiden. Ein Kennzahlensystem besteht aus der Kennzahlenmenge und einer Kennzahlenstruktur, die Struktur ihrerseits aus einer Mehrzahl von Substrukturen. In der restlichen Literatur fehlt die Unterscheidung zwischen Struktur und System vollständig, es existiert sogar keine Teilung der Gesamtstruktur in Substrukturen. Dort wird also über Rechen- oder Planungssysteme gesprochen und nicht von Rechen- und Planungsstrukturen.

Die Erklärung dafür ist einfach - die jeweiligen Autoren entwickeln konkrete Systeme, die z.B. nur rechnerische Zusammenhänge enthalten und nur für Planung bestimmt sind. Da in dem Gesamtsystem nur eine Struktur existiert, besteht keine Notwendigkeit in der Unterscheidung zwischen System und Struktur. Ein reales Kennzahlensystem wird in der Regel für alle fünf Führungsaufgaben verwendet und enthält eine große Anzahl von unterschiedlich konzipierten Teilstrukturen, deshalb ist es in einer praktischen Arbeit zwingend notwendig diese Unterscheidung zu machen.

2.1.5 Strukturierung der Kennzahlensysteme

In diesem Abschnitt vertiefen wir die Diskussion des Klassifizierungsmerkmals "Strukturierungsart", das im vorherigen Abschnitt vorgestellt wurde.

1) Rechenstrukturen

Rechenstrukturen werden folgendermaßen definiert ([März83], S. 65):

Rechenstrukturen sind Kennzahlenstrukturen, in denen jede Kennzahl in rechentechnischer Verknüpfung mit anderen Kennzahlen steht. Dadurch entsteht eine hierarchisch aufgebaute Pyramide, an deren Spitze eine hoch verdichtete Spitzenkennzahl steht.

Rechentechnisch verknüpft sind die Kennzahlen dann, wenn sich eine Kennzahl durch rechnerische Methoden aus den restlichen Kennzahlen entwickeln läßt ([Meye94])

In dieser Arbeit modellieren wir die Kennzahlenstruktur mit Graphen, die aus Knoten und Kanten bestehen. In Rechenstrukturen gibt es zwei Typen von Knoten: Kennzahlenknoten und Rechenknoten. Die Kennzahlenknoten stellen die Kennzahlen dar, die Rechenknoten symbolisieren die Berechnungsvorschriften der Kennzahlenfunktionen. Die eventuellen funktionalen Zusammenhänge zwischen den Kennzahlen(-funktionen) werden durch die Kanten ausgedrückt (Abbildung 2-2 Rechensubstruktur).

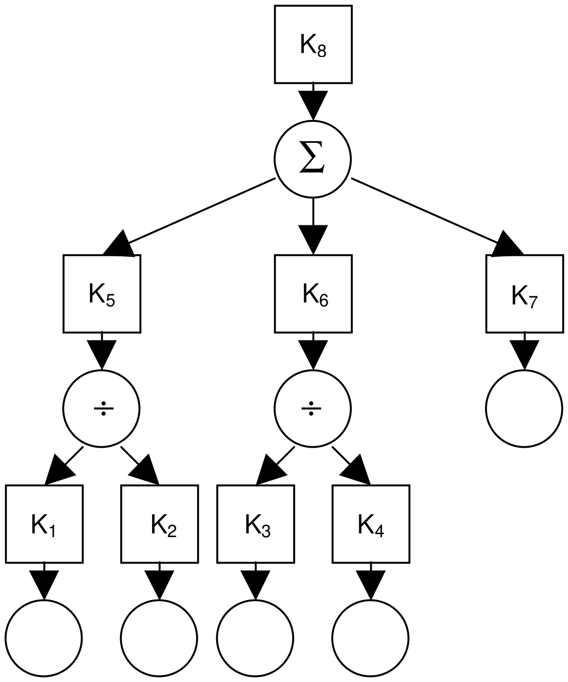


Abbildung 2-2 Rechensubstruktur

In der Abbildung 2-2 sind die Kanten des Graphen durch Pfeile repräsentiert, der Graph ist also gerichtet. Hätte ein ungerichteter Graph nicht ausgereicht? Nun, es gibt mehrere Gründe dafür, daß der Graph gerichtet sein muß. Der Knoten K_5 aus derselben Abbildung ist z.B. mit zwei Rechenknoten verbunden (einem (Σ)-Knoten und einem (÷)-Knoten; der (÷)-Knoten ist der Knoten mit der Berechnungsvorschrift der Kennzahl). Bei einem ungerichteten Graphen könnte kein Algorithmus mehr unterscheiden, welcher Rechenknoten der besondere Knoten mit der Berechnungsvorschrift ist. Das, was auf einer Zeichnung für den Entwickler der Kennzahlenstruktur sofort erkennbar ist, kann ein Algorithmus nicht ohne weiteres bestimmen. Ähnliche Probleme treten auch bei den Ordnungsstrukturen auf, die wir im nächsten Abschnitt behandeln. Das Framework verwendet deshalb gerichtete Graphen zur Modellierung der Kennzahlenstrukturen.

Die Kennzahl K_8 in der Abbildung 2-2 ist die sog. Spitzenkennzahl der Substruktur, sie wird funktional aus K_5, K_6 und K_7 durch die Summenbildung zusammengesetzt. Die Kennzahlen K_1, K_2, K_3, K_4, und K_7 sind terminal, d.h. sie basieren auf keinen anderen Kennzahlen, sondern beziehen die Daten direkt aus den Datenquellen, wie z.B. Performance-Statistiken, Trouble-Tickets-Datenbanken und ähnlichem (mehr darüber im Kapitel 3). Die Kennzahlen K_8, K_5, K_6 in der Abbildung sind nichtterminal. Die Rechenknoten der terminalen Kennzahlen (durch leere Kreise dargestellt) bilden die unterste Ebene des Kennzahlengraphen.

Die Verwendung von gerichteten Graphen ist unumgänglich, über die Richtung der Kanten kann allerdings diskutiert werden. Die gezeigte top-down Richtung drückt den funktionalen Zusammenhang der Kennzahlen aus. Die Spitzenkennzahl K_8 hängt von den Kennzahlen K_5, K_6

und K_7 ab, deshalb sich die Kanten zu diesen gerichtet. Alternativ könnte die bottom-up Richtung ausgewählt werden, die dem Datenfluß entspricht : die Daten fließen von den Datenquellen (in der Abbildung nicht eingezeichnet) über terminale und nichtterminale Kennzahlen bis hin zu der Spitzenkennzahl. Beide Alternativen sind plausibel, keine von ihnen hat besondere Vorteile oder Nachteile. Wir wählten für unser Framework die top-down Richtung.

2) Ordnungsstrukturen

Die Definition von Ordnungsstrukturen übernahmen wir aus ([März83], S.68):

Unter Ordnungsstrukturen[13] versteht man Kennzahlenstrukturen, zwischen deren Elementen (den Kennzahlen, Bem. des Autors) keine rechnerische Beziehung besteht, wohl aber ein sachlogischer Zusammenhang gegeben ist.

Ordnungsstrukturen teilen die zugrundeliegenden Kennzahlen in verschiedene Gruppen ein (z.B nach Managementbereichen oder nach Benutzergruppen). Im Prinzip gibt es keine Grenzen für die Tiefe und Breite der Unterteilung, man sollte aber die Bedienbarkeit durch die Benutzer des Kennzahlensystems nicht vernachlässigen.

Ähnlich wie bei den Rechenstrukturen, modellieren wir die Ordnungsstrukturen durch gerichtete Graphen. In einer Ordnungsstruktur gibt es zwei Typen von Knoten: Kennzahlenknoten und Ordnungsknoten. Die Ordnungsknoten entsprechen den Differenzierungskriterien, anhand derer man die Kennzahlen in unterschiedliche Gruppen einteilt; die Kanten sind mit Werten für diese Kriterien markiert. Alle Kennzahlenknoten befinden sich auf der untersten Ebene der Struktur.

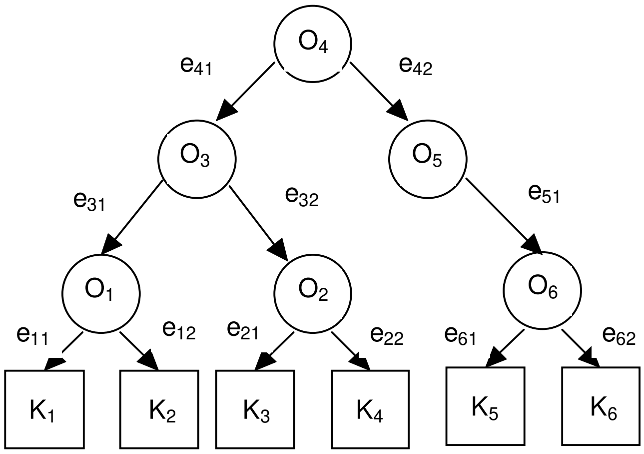


Abbildung 2-3 Ordnungssubstruktur

Beispiele für Ordnungsstrukturen findet man weiter unten in Abschnitten 2.2, 2.3 und 2.4.

3) Heuristische Kennzahlenstrukturen

Heuristisch strukturiert sind Kennzahlensysteme, die von einem Anwender zur Lösung eines bestimmtes Problems aufgestellt werden und ausschließlich solche Kennzahlen enthalten, die nach dessen subjektivem Empfinden zur Beschreibung des Sachverhalts erforderlich sind. Die Auswahl der Kennzahlen erfolgt subjektiv (Definition angelehnt an [März83], S. 69).

[13] Im Original steht hier : "Unter Ordnungssystemen…".

Man kann diese Strukturen durch Graphen modellieren, wo alle Kennzahlenknoten mit einem einzigen Zusammenhangsknoten verbunden sind (Abbildung 2-4).

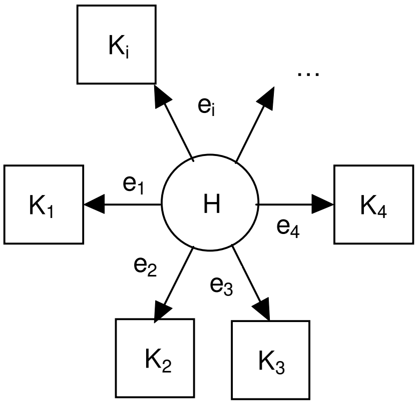


Abbildung 2-4 Heuristisches Kennzahlensubsystem

Heuristische Strukturen sind, gemessen an der Verflechtung der Struktur, die einfachsten im Vergleich mit anderen drei Strukturtypen. Außerdem sind sie auch deutlich "kurzlebiger" - infolge der Zeit sammelt der Anwender eine Erfahrungsbasis, anhand derer die heuristische Struktur entweder verworfen, oder aber zu einer der drei "langlebigen" Strukturen wird. Kann der Benutzer mit dem gesammelten Datenmaterial belegen, daß die Kennzahlen in einem statistischen Zusammenhang stehen, so erfolgt die Definition einer empirisch-statistischen Substruktur. Häufig wird das ursprüngliche heuristische System sachlogisch strukturiert - als Ergebnis erhält man eine Ordnungsstruktur. Theoretisch möglich ist auch die Konvertierung zu einem Rechensystem, wenn der beobachtete statistische Zusammenhang eine hundertprozentige Korrelation hat (zur Erklärung der statistischen Zusammenhänge siehe einige Zeilen weiter unten).

d) Empirisch-statistische Kennzahlenstrukturen

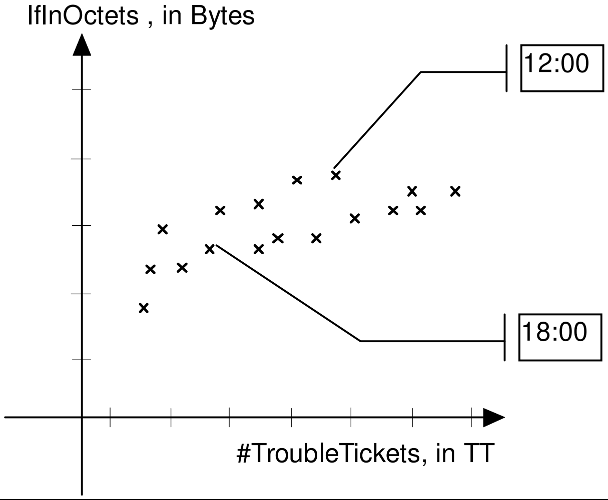
Empirisch-statistische Kennzahlenstrukturen sind Strukturen, die aufgrund empirischer Untersuchungen im Zusammenhang mit bestimmten Fragestellungen ermittelt worden sind. Der Unterschied zu heuristisch strukturierten Systemen besteht darin, daß die Bedeutung der Kennzahlen-Zusammenstellung für die Vorhersage oder Beurteilung eines bestimmten Sachverhaltes empirisch untersucht und statistisch belegt ist. Die Auswahl der Kennzahlen erfolgt somit objektiv (Definition übernommen aus [März83], S. 69 ff.).

Bei den empirisch-statistischen Untersuchungen handelt es sich um die Aufstellung von Hypothesen der Form "Die Kennzahlen K_1 bis K_n korrelieren miteinander (sind abhängig)" und anschließende Überprüfung der Aussagen anhand der vorhandenen Daten. Dieser Vorgang heißt in der modernen Informatik-Sprache auch Data Mining. Wir geben im folgenden eine kurze Beschreibung der Vorgehensweise bei diesen Untersuchungen (entnommen aus [GrTi95]):

- Ausgangssituation

  Eine k-dimensionale "Meßreihe" aus n Messungen $(x_1,y_1,\ldots)$, …, $(x_n,y_n,\ldots)$ ist gegeben. Diese Meßreihe besteht im Falle eines Kennzahlensystems aus Werten von Kennzahlen z.B. zu unterschiedlichen Zeitpunkten. Beispiel:

Bei der Korrelationsanalyse von zwei Kennzahlen ifInOctets und #TroubleTickets erhält man eine Punktwolke im x-y-Diagramm. Die Meßreihe ist zweidimensional, die Punkte stellen die Kombinationen der Kennzahlenwerte zu unterschiedlichen Zeitpunkten dar.



- Es sind folgende Fragen möglich:
  1) Korrelieren die Kennzahlenwerte in irgendeiner Weise miteinander ? Bezogen auf unsere Meßreihe formuliert sich die Frage als: "Gruppieren sich die Punkte um eine Kurve bzw. eine k-dimensionale Fläche (z.B. eine Gerade, Parabel, Kegel,…)?".
  2) Wie kann eine derartige Kurve ermittelt werden?

- Vorgehensweise bei der Beantwortung der Fragen:
  1) Es wird eine Hypothese über einen bestimmten funktionalen Zusammenhang (bzw. die Form der Kurve) aufgestellt. Konkret heißt das , daß der Kurventyp vorgeschlagen wird (Gerade, Parabel, Exponential-Funktion usw.).
  2) Es werden die Parameter der Kurve des geforderten Kurventyps berechnet, die die Minimierung des Abstandes zwischen den Datenpunkten und der Kurve sicherstellen. Im Falle von Geraden bedeutet das eine Berechnung der Koeffizienten a und b aus der Gleichung y = ax+b.
  3) Die Schritte 1 und 2 werden solange wiederholt, bis die Unterschiede zwischen der Punkt-"Wolke" und der Kurve (ausgedrückt durch die Summe der Abstände der Datenpunkte von der Kurve) auch über alle möglichen Kurventypen minimal ist. Dabei wird auf die Einfachheit der Kurven/Funktionen geachtet - einfache Funktionen wie z.B. "f(x)=2x+100" sind vorzuziehen, auch wenn sie nicht das absolute Optimum darstellen.

Den möglichen Nutzen dieser mathematischer Darbietungen für ein Rechenzentrum zeigen wir an einem Beispiel[14]:

[14] Dieses Beispiel ist fiktiv, ihm ist keine reale Untersuchung zugrundegelegt. Die Aussagen und die vorgeführte Korrelation sind allerdings tendenziell richtig: die Verfügbarkeit kann den Wert 100 % nicht überschreiten, ihre Erhöhung "kostet" in dem oberen Wertebereich unproportional mehr, als bei mittleren oder niedrigen Ausgangswerten, die HTTP-Server brauchen einen Minimalumfang von Speicher, um überhaupt verfügbar zu sein. Die hypothetische Korrelationsfunktion erfüllt alle diese Bedingungen.

Durch eine Untersuchung sei eine Korrelation zwischen zwei Kennzahlen "Hauptspeicherausstattung einer Workstation" und "Verfügbarkeit des HTTP-Servers auf einer Workstation" festgestellt. Der Zusammenhang erweise sich annähernd als umgekehrt quadratisch proportional:

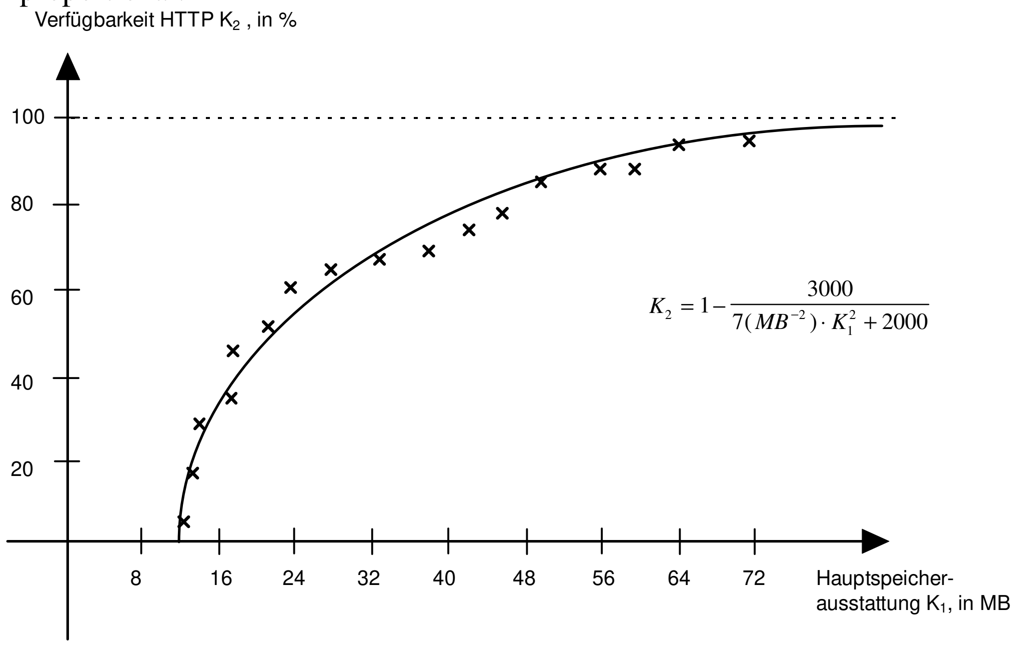


Abbildung 2-5 Hypothetische Korrelation zwischen Hauptspeicherausstattung und Verfügbarkeit

Die Kreuze in der Abbildung sind die Ergebnisse einer Stichprobe. Die Kurve ist eine statistisch optimierte Annäherung. Als Ergebnisse der Untersuchung könnte man z.B. folgende Erkenntnisse betrachten:

- Die Erhöhung der Verfügbarkeit von 55 % auf 75 % "kostet" 14 MB an zusätzlichen Hauptspeicher,
- Die gleichhohe Erhöhung (20 %), aber von 75 % auf 95%, erfordert bereits 30 MB von neuen Speichermodulen.

Falls ein Rechenzentrum mehrere Dutzend Workstations unterhält, die als HTTP-Server dienen, und eine Erhöhung der Verfügbarkeit erwünscht ist, so kann eine ungefähre Anzahl von zu beschaffenden Speichermodulen relativ leicht errechnet werden.

Die Untersuchungen setzen die Existenz eines Kennzahlensystems nicht voraus. Es ist aber sehr nützlich, den Vorgang der Validierung einer Hypothese durch ein automatisiertes System zu unterstützen. Erst dann werden auch weitgehende Untersuchungen möglich[15]. Aus diesem Grund bauen wir die Unterstützung der empirisch-statistischer Kennzahlenstrukturen in unser Framework ein. Die von einem Kennzahlensystem angebotene Unterstützung könnte z.B. aus der automatischen Durchführung des zweiten Schrittes, also der Kurvenbestimmung, bestehen. Einige weitere Erleichterungen für den Benutzer können auch für die Stufen 1 (Vorauswahl des Kurventyps) und 3 (iterative Optimierung des Ergebnisses) realisiert werden. Der Benutzer bräuchte dann zum Beginn der Untersuchung nur die zu untersuchenden Kennzahlen plus vermutete Zusammenhangstypen angeben. Die Berechnung und die Zeichnung der empirisch-statistischen Struktur sollte das System selbständig durchführen können.

[15] Die Anzahl der zu überprüfenden Hypothesen ist selbst bei einer Beschränkung auf 2-Kennzahlen-Korrelationen und ausschließlich lineare Zusammenhänge bereits quadratisch proportional der Anzahl der Kennzahlen im Kennzahlensystem.

Graphen, die empirisch-statistische Strukturen modellieren, enthalten Kennzahlenknoten und Knoten, die empirisch-statistische Zusammenhänge repräsentieren. Die Kanten verbinden die Knoten miteinander (Abbildung 2-6).

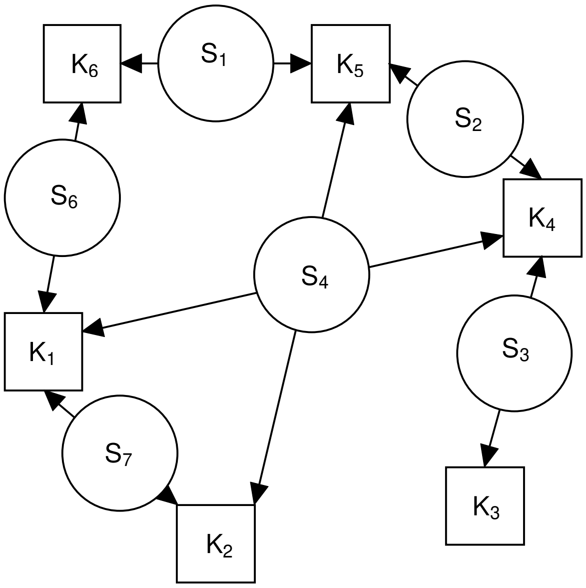


Abbildung 2-6 Empirisch-statistische Kennzahlensubstruktur

Die maximale Anzahl von Kennzahlenknoten, die mit den S-Knoten verbunden werden, hängt von der Art der empirisch-statistischen Untersuchung ab, in der die Zusammenhänge festgestellt wurden. Falls man nur die Korrelation von Kennzahlenpaaren untersucht hat, so sind auch genau zwei K-Knoten durch ein S-Knoten verbunden.

☑

Die vollständige Kennzahlenstruktur eines realen Kennzahlensystems kann und sollte eine Vielzahl von verschiedenartigen Substrukturen enthalten (Abbildung 2-7). Es empfiehlt sich lieber eine zusätzliche, an die Anforderungen eines bestimmtes Benutzers zugeschnittene Struktur zu definieren, als ihn mit einer Standardstruktur arbeiten zu zwingen. Die Komplexität des Systems, das aus einer Vielzahl von Strukturen besteht, kann gegenüber dem Benutzer dadurch reduziert werden, daß ihm nur die für ihn relevanten Teile der Gesamtstruktur angezeigt werden.

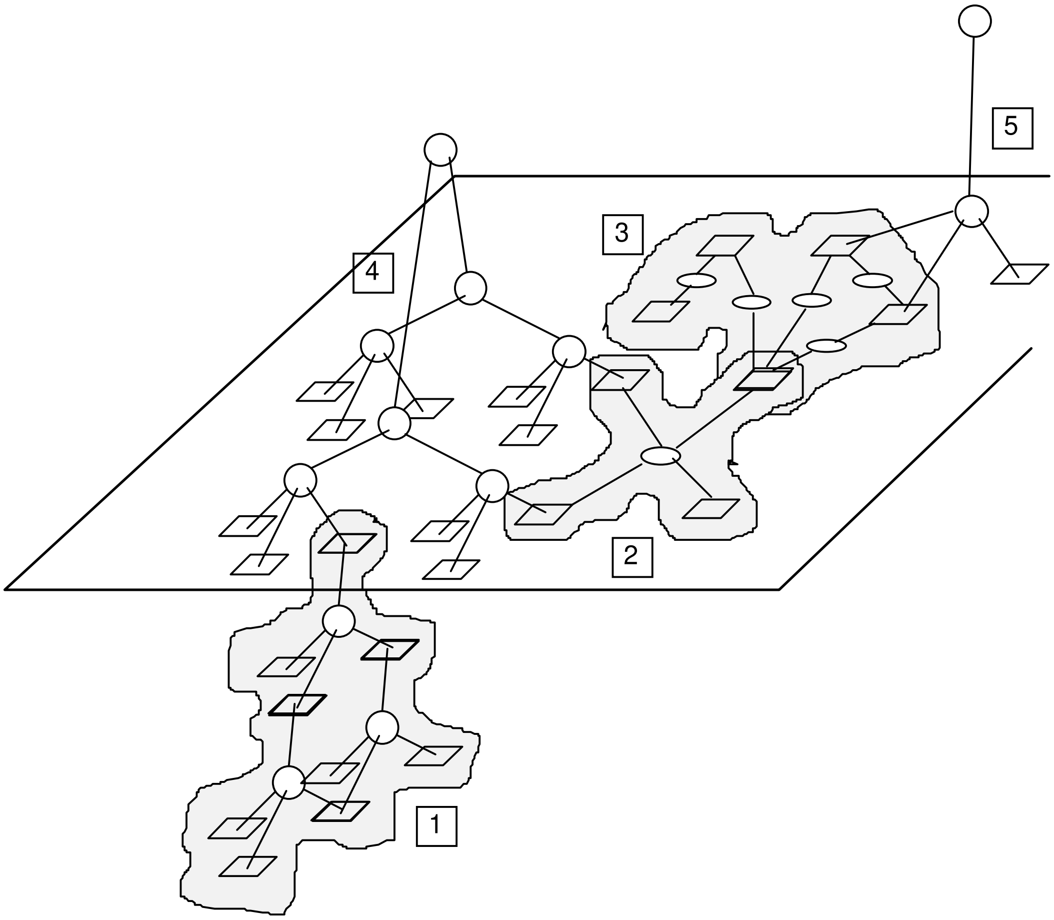


Abbildung 2-7 Beispiel eines Kennzahlensystems

In der Abbildung findet man zwei Ordnungsstrukturen (Nr.4 und Nr.5) und jeweils eine Rechenstruktur(Nr. 1), heuristische Struktur (Nr. 2) und empirisch-statistische Struktur (Nr. 3). Zur besseren Unterscheidung haben wir die Strukturen Nr.1, 2 und 3 durch Schattierung hervorgehoben. Wie man gut erkennen kann, überlappen sich die Substrukturen in mehreren Kennzahlen.

2.1.6 Einordnung der Kennzahlensysteme in betriebliche Führungsaufgaben

Wir fahren mit der Diskussion der Klassifizierungsmerkmale, die im Abschnitt 2.1.4 begonnen wurde, fort und stellen in diesem Abschnitt die “Verwendungsorientierung” vor. Zuerst aber möchten wir ein allgemeines Modell der betrieblichen Führungs- und Ausführungsaufgaben (vgl. [Meye94], S. 13 ff.) vorführen.

Der Ablauf des betrieblichen Geschehens läßt sich als ein Prozeß auffassen. In seiner einfachsten Darstellung können folgende Handlungsphasen unterschieden werden:

a) Planung (Zielbestimmung, Problemerkennung, Planung i.e.S.),
b) Durchführung (Realisierung der "Planung i.e.S.", Zwischenkontrolle),
c) (End-)Kontrolle (Vergleich des Geplanten mit dem Erreichten, Abweichungsanalyse, Betriebsvergleiche).

Zur Unterstützung dieser Handlungsphasen benötigt ein Betrieb Informationen. Die Kennzahlen sind eine besondere Art der verwendeten Informationen, nämlich quantifizierte Informationen.

Entsprechend diesem einfachen Modell können Kennzahlen und Kennzahlensysteme in allen drei Stufen der Führung und Ausführung eingesetzt werden. Wir geben im folgenden eine Reihe von Eigenschaften an, die nützlich für die Beschaffung von Informationen in den jeweiligen Phasen sind.

1) <u>Planung</u>

Bei der Planung sind die "Was-wäre-wenn"- Fragen ganz wichtig. Schließlich will man wissen, wie die geplanten Aktionen sich auf den Betrieb auswirken. Interessant wäre z.B. zu wissen, wie die Veränderung einer Größe um X% (z.B. Prozessorleistung eines FTP-Server) sich auf eine andere Größe auswirkt (z.B. Verfügbarkeit des FTP-Dienstes).

Diese Frage kann bei expliziten rechentechnischen Zusammenhängen auf die Elastizität einer Kennzahl in Abhängigkeit von einer anderen Kennzahl abgebildet werden:

$$\mathit{Elastizität}\left[K_1, K_2\right] = \frac{\partial K_1}{\partial K_2} \cdot \frac{K_2}{K_1}$$

Die Elastizität ist eine Funktion, die auf demselben Wertebereich definiert ist, wie auch die Kennzahlen K_1 und K_2 . Sie besagt, wie “elastisch” eine Funktion auf die Veränderungen einer anderen Funktion reagiert. Dazu ein Beispiel:

> Seien K_1 und K_2 Kennzahlen(-funktionen) und gelte ferner $K_1 = ½\ K_2$. Die Elastizitätsfunktion berechnet sich wie folgt:
>
> $$\mathit{Elastizität}\left[K_1, K_2\right] = \frac{\partial K_1}{\partial K_2} \cdot \frac{K_2}{K_1} = \frac{1}{2} \cdot \frac{K_2}{\frac{1}{2} K_2} = 1$$
>
> Sie ist also die 1-Funktion - für beliebige Parameterwerte hat sie den Wert 1. Die Konsequenz dieser Analyse ist, daß eine X-prozentige Veränderung der Kennzahl K_2 sofort eine gleichhohe Veränderung der Kennzahl K_1 nach sich zieht, und zwar unabhängig davon, welcher Wert der Kennzahl K_2 als Basis der Veränderung dient.

Zur Unterstützung der Elastizitätsanalyse durch ein Kennzahlensystem ist eine Rechenstruktur notwendig. Durch die Berücksichtigung von allen vorhandenen Rechenzusammenhängen, können Analysen auch von nicht unmittelbar abhängigen Kennzahlen vorgenommen werden.

Falls keine expliziten funktionalen Zusammenhänge bekannt sind, so können empirisch-statistische Untersuchungen durchgeführt werden. Die Ergebnisse dieser Untersuchungen sind Funktionen, die so optimiert sind, daß die tatsächlichen Werte der Kennzahlen nur minimal von den Werten der Funktionen sich unterscheiden. Die Möglichkeit des Einsatzes von emp.-stat. Kennzahlenstrukturen bei der Analyse von Kennzahlen-Abhängigkeiten haben wir in einem Beispiel im vorherigen Abschnitt diskutiert.

Eine andere mögliche “Was-wäre-wenn”-Frage formuliert sich folgendermaßen. Es werden Sollwerte (bzw. Soll-Funktionen) für eine oder mehrere Kennzahlen vorgegeben, das Kennzahlensystem soll anhand dieser Sollwerte und mit der Default-Besetzung anderer Kennzahlen (z.B. Besetzung mit Vorperiode-Istwerten) bestimmte Kennzahlen berechnen.

Das Framework löst diese Aufgabe dadurch, daß die Sollwerte einfach in die Funktionsvorschriften der zu berechnenden Kennzahlen eingesetzt werden.

Sei eine Rechenstruktur wie in der Abbildung 2-8 gegeben. Der Anwender interessiere sich für die möglichen Werte der Kennzahl K_8 , falls die Kennzahlen K_1 und K_2 bestimmte (Soll-)Werte annehmen. Alle anderen Kennzahlen, von denen K_8 explizit oder implizit abhängt, sollen als unverändert betrachtet werden, d.h. es sind die Istwerte z.B. des vorigen Tages zu nehmen.

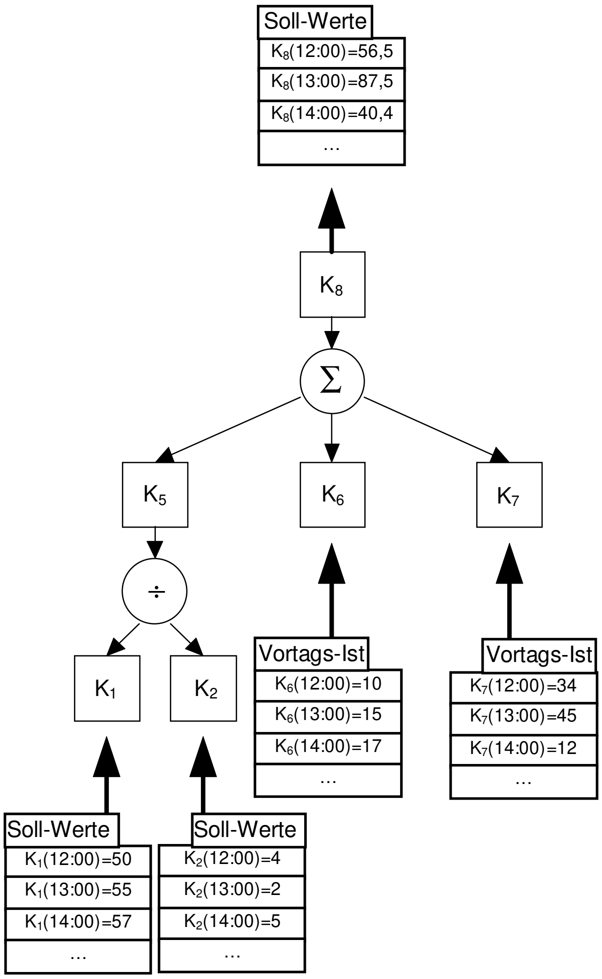


Abbildung 2-8 Unterstützung der Planung durch Rechenstrukturen

Der Prozeß der Analyse sieht folgendermaßen aus. Der Benutzer definiert mit einem geeigneten Werkzeug die Sollwerte der zwei Kennzahlen K_1 und K_2. Diese Daten fließen dann in den üblichen Datenstrom einer Kennzahlenberechnung. Das Ergebnis (die Sollwerte der Kennzahl K_8) werden mit einem graphischen Programm z.B. in der Form einer Kurve dargestellt. Die Auswirkungen der Veränderungen der Sollwerte von K_1 und K_2 auf die Kennnzahl K_8 werden dem Benutzer sofort angezeigt.

2) Endkontrolle (Abweichungsanalyse)

Bei der Endkontrolle allgemein und bei der Abweichungsanalyse insbesondere werden keine Echtzeitwerte gebraucht. Wichtig sind hier Gegenüberstellungen von Soll- und Ist-Werten, es werden somit Abweichungskennzahlen benötigt. Deshalb sollte das Kennzahlensystem um eine Struktur erweitert werden, die diese Differenz-Kennzahlen enthält. Dabei lassen sich allgemeine Kennzahlen-Templates bilden, die auf jede Kennzahl anwendbar sind, wie z.B.:

- $\Delta_{rel}^{soll} K$ - relative Abweichung einer Kennzahl K vom Sollwert zum Zeitpunkt t, definiert durch

$$\Delta^{soll}_{rel} K(t) := \frac{K(t,"Soll") - K(t,"Ist")}{K(t,"Soll")}$$

- $\Delta^{soll}_{abs} K$ - absolute Abweichung einer Kennzahl K vom Sollwert zum Zeitpunkt t, definiert durch

$$\Delta^{soll}_{abs} K(t) := K(t,"Soll") - K(t,"Ist")$$

Falls beunruhigende Abweichungen festgestellt worden sind, müssen "Ursache-Wirkung"- Zusammenhänge leicht erkennbar sein. Die Rechen- und empirisch-statistische Strukturen bieten per Konzeption eine Unterstützung für die Analyse der "Ursache-Wirkung"-Abhängigkeiten. In Rechenstrukturen ist jede nichtterminale Kennzahl von einer bestimmten Menge von anderen Kennzahlen abhängig (terminale Kennzahlen basieren entsprechend auf einer Menge von Datenquellen). Die Ursache der Abweichungen kann somit nur in den Entwicklungen dieser Basiskennzahlen liegen.

Die Zuordnung der Ursachen und Wirkungen ist bei empirisch-statistischen Strukturen nicht eindeutig. Trotzdem bieten sie gute Anhaltspunkte, falls die zugrundegelegte empirisch-statistische Untersuchung adäquat durchgeführt wurde. In jedem Fall sind dann nur abgeschwächte Aussagen möglich, wie z.B. : "Die Ursache für die Abweichung der Verfügbarkeit eines Routers von den Sollwerten liegt mit 80 % Wahrscheinlichkeit bei dem erhöhten Datendurchsatz in den letzten 3 Monaten".

Die Ordnungsstrukturen sind nur dann zur Feststellung der Abweichungsgründen geeignet, wenn die Ordnungsstruktur explizit aus einer "Ursache-Wirkung"- (bzw. "Ziel-Mittel"-) Hierarchie übernommen wurde. Wir zeigen im Abschnitt 2.2 Substruktur "Ziel-Mittel-Hierarchie des Rechenzentrums, wie eine solche Ordnungsstruktur für ein Rechenzentrum aufgebaut werden kann.

3) Steuerung

Im Unterschied zu der Endkontrolle werden bei der Steuerung überwiegend Istwerte gebraucht. Es werden aber auch Gegenüberstellungen von Istwerten mit Sollwerten und von Istwerten mit Vorperioden-Istwerten benötigt. Ähnlich wie bei Abweichungsanalyse, lassen sich allgemeine Kennzahlen-Templates bilden:

- $\Delta^{vor}_{rel} K$ - relative Veränderung einer Kennzahl K bzgl. des Vorperiode-Wertes zum Zeitpunkt t, definiert durch

$$\Delta^{vor}_{rel} K(t) := \frac{K(t,"Ist") - K(t-\Delta t,"Ist")}{K(t,"Ist")}$$

- $\Delta^{vor}_{abs} K$ - absolute Veränderung einer Kennzahl K bzgl. des Vorperiode-Wertes zum Zeitpunkt t , definiert durch

$$\Delta^{vor}_{abs} K(t) := K(t,"Ist") - K(t-\Delta t,"Ist")$$

Als Beispiel für die Anwendung eines solchen Template läßt sich die Steuerung der Verfügbarkeit angeben:

Die relative Veränderung der Plattformverfügbarkeit wird entsprechend dem Template definiert:

$$\Delta_{rel}^{vor} Plattformverfügbarkeit(p,t) := \frac{Plattformverfügbarkeit(p,t,"Ist") - Plattformverfügbarkeit(p,t-\Delta t,"Ist")}{Plattformverfügbarkeit(p,t,"Ist")}$$

Falls z.B. am 31.12.1997 die Verfügbarkeit der Plattform “sun1” 95 % beträgt, einen Tag davor sie aber 100 % war, so fand am 31.12.1997 eine negative Veränderung der Verfügbarkeit um 5,3 % statt:

$$\Delta_{rel}^{vor} Plattformverfügbarkeit(sun1, 31.12.1997) = \frac{95\% - 100\%}{95\%} = -5{,}3\%$$

An die Kennzahlenstrukturen, die zur Unterstützung der operativen Steuerung des Rechenzentrums eingesetzt werden sollen, werden besondere Anforderungen gestellt. Die Mitarbeiter des Rechenzentrums, die für die operative Steuerung verantwortlich sind, brauchen technische Informationen, die detailliert und aktuell sind und die den Zustand widerspiegeln, in dem die Technik sich momentan befindet (z.B. Anzahl fehlerhafter Pakete, die aus einem Router verschickt werden, oder aktuelle CPU-Auslastung eines Servers). Diese Informationen werden benötigt, um kurzfristige Entscheidungen treffen zu können (z.B. Reboot des Routers oder Terminierung eines Benutzer-Programms, das die Server-CPU inakzeptabel belastet). Aus diesem Grund benötigen sie eine Struktur , in der die technischen Objekte des Rechenzentrums mit jeweiligen Charakteristika eingeordnet sind. Diese Idee wird im Abschnitt 2.3 Substruktur “Technische Aspekte des Rechenzentrums weiterentwickelt.

2.2 Substruktur “Ziel-Mittel-Hierarchie des Rechenzentrums”

Mit dem letzten Unterabschnitt ist unsere Vorstellung der theoretischen Grundlagen von Kennzahlensystemen abgeschlossen und wir beginnen mit der Definition von konkreten Kennzahlenstrukturen.

Der Bedarf an Kennzahlen ergibt sich aus den zu lösenden konkreten betrieblichen Aufgaben. Deshalb wird von vielen Autoren (vgl. [Meye94], [Hein76]) vorgeschlagen zuerst mit der Formulierung dieser konkreten Aufgaben zu beginnen (Aufbau der sog. Ziel-Mittel-Hierarchie). Während des Führungsprozesses wird in der Regel von einem zu erreichenden Oberziel ausgegangen, von dem sich mit Hilfe des Ziel-Mittel-Prinzips primäre, sekundäre, tertiäre usw. Unterziele ableiten lassen. Wenn sich kein absolutes Oberziel bestimmen läßt, dann kann die Ziel-Mittel-Hierarchie an der Spitze mehrere gleichgewichtige Oberziele beinhalten.

Die Aufstellung dieser Hierarchie liefert eine Ordnungsstruktur, die sich gut zur Endkontrolle eignet, da die Erreichung der Ziele anhand der Kennzahlen überprüft werden kann. Sie läßt sich auch zur Planung (die Planung kann aus der Formulierung einer solchen Hierarchie bestehen) und Abweichungsanalyse (die Nichterreichung der Ziele kann zurückverfolgt werden) einsetzen. Das Subsystem ist per Konzept ausschließlich auf das Management des Rechenzentrums ausgerichtet.

Die Blätter (die Kennzahlenknoten) des Ziel-Mittel-Graphen müssen weiter spezifiziert werden. Dazu wird vorgeschlagen für jedes Blatt ein geeignetes Rechensystem aufzustellen. Dieses erlaubt dann auch die Überwachung des Erreichung der atomaren Ziele.

In dieser Arbeit werden idealtypische Ziel-Mittel-Hierarchien für zwei Arten von Rechenzentren vorgestellt - für kommerzielle (Abbildung 2-9) und für gemeinnützige (Abbildung 2-11) Rechenzentren. Für Cost Center und Service Center Rechenzentren ist der Aufbau der Hierarchien entsprechend vorzunehmen. Wir weisen ausdrücklich darauf hin, daß in konkreten Rechenzentren die Hierarchien deutlich komplexer sind, als hier angegeben. So kann ein Universitäts-Rechenzentrum durchaus das Ziel “Erhöhung der Marktanteile” verfolgen. In der heutigen Zeit der Dezentralisierung der Informationstechnologie müssen auch solche Rechenzentren um ihre Kunden kämpfen, sonst droht ihnen der Entzug der Existenzbegründung. Umgekehrt können auch kommerzielle Rechenzentren die “Einhaltung des Budgets” als eines der primären Ziele betrachten und dementsprechend agieren.

Bei kommerziellen Rechenzentren gehen wir von zwei primären Zielen aus - der Steigerung des Umsatzes und der Minderung der Kosten. Zusammen betrachtet, sind das die Mittel des Oberziels “langfristige Gewinnmaximierung”.

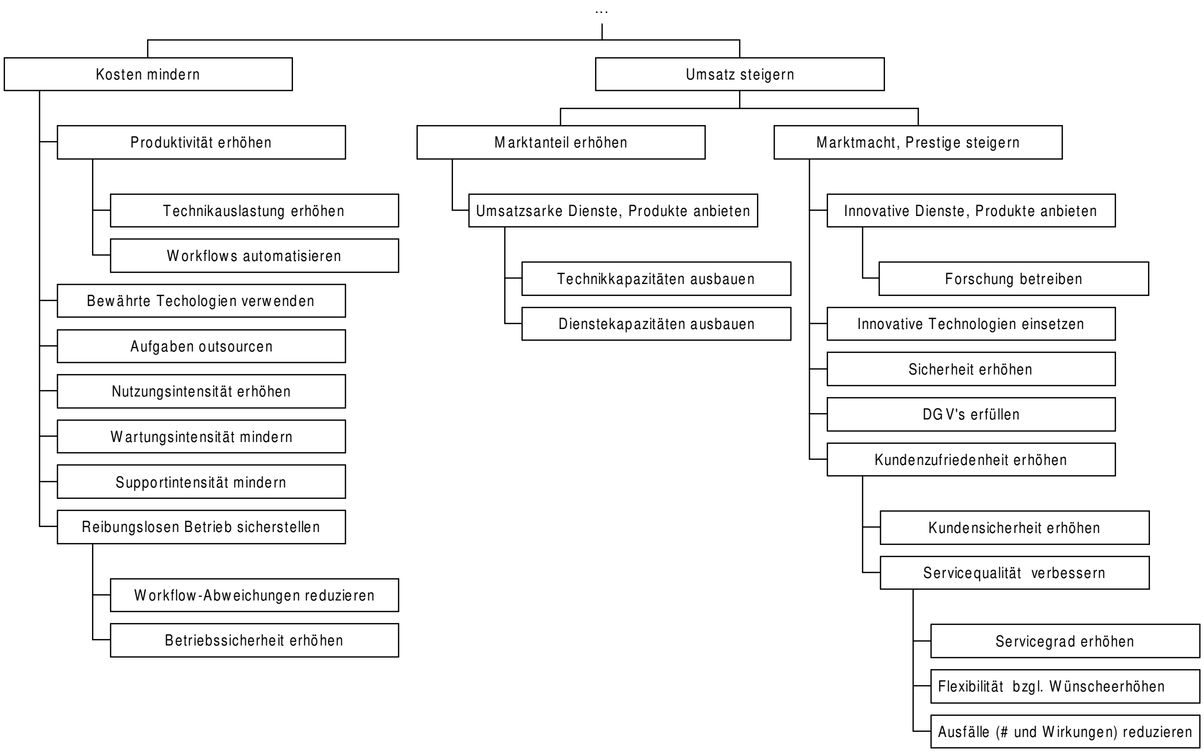


Abbildung 2-9 Ziel-Mittel-Hierarchie für kommerzielle Rechenzentren

Technische Unterziele tauchen erst ab der zweiten Stufe der Ziele-Hierarchie (Abbildung 2-9) auf. Im einzelnen sind das die:

- Erhöhung der Technikauslastung,
- Automatisierung der Workflows,
- Erhöhung des Anteiles der bewährten Technologien,
- Erhöhung der Nutzungsintensität,
- Minderung der Wartungsintensität,
- Minderung der Supportintensität,
- Reduktion der Workflow-Abweichungen,
- Erhöhung der Betriebssicherheit,
- Ausbau der Technik- und Dienstkapazität,
- Erhöhung des Anteiles der innovativen Technologien,
- Verbesserung des Grades der DGV[16]-Erfüllung,
- Erhöhung der Kundensicherheit,
- Erhöhung der Servicegrades,
- Reduktion der Anzahl der Ausfälle, Erhöhung der Verfügbarkeit.

[16] DGV - Dienstgütevereinbarungen.

Diese technische Hierarchie ist nur ein Teil der allgemeinen Ziel-Mittel-Hierarchie der Unternehmensart “kommerzielles Rechenzentrum”. Aus diesem Grund haben wir bei der Definition der obersten zwei Stufen uns eng an das Mayer’sche “Mittel-Zweck-Schema für Unternehmen” gehalten (Abbildung 2-10).

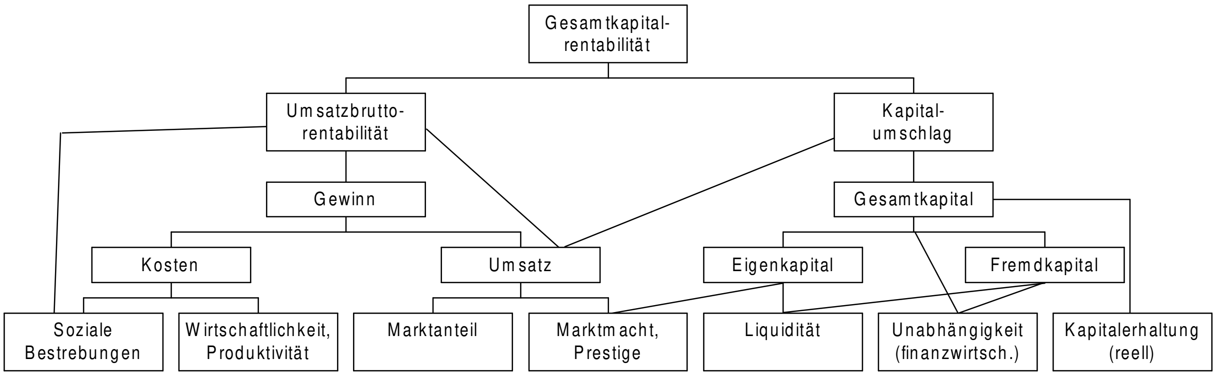


Abbildung 2-10 Deduktiv orientiertes Mittel-Zweck-Schema (nach Mayer und Heinen)

Nur für die Ziele “Kosten mindern” und “Umsatz steigern” aus der Abbildung 2-10 lassen sich technische Mittel (“Mittel” im Sinne des Ziel-Mittel-Prinzips) einsetzen, andere “Äste” in der Abbildung 2-10 sind entweder rein betriebswirtschaftlicher, oder rein sozialer Natur.

Einige Ziele sind entgegengerichtet, wie z.B. die “Erhöhung des Anteiles der bewährten Technologien” und die “Erhöhung des Anteiles der innovativen Technologien”, die zur Kostenminimierung bzw. zur Umsatzmaximierung eingesetzt werden. Die richtige Gewichtung der Ziele stellt eine Herausforderung für die Führung des Rechenzentrums.

Der Vorgang der Aufspaltung der Ziele in die Mittel wird solange fortgesetzt, bis ein Grad der Operationalität von Zielen erreicht ist , der eine unmittelbare Realisierung durch ein Mittel ermöglicht. Für die Ordnungsstruktur bedeutet das, daß hier der letzte Ordnungsknoten anzubringen ist. An ihn werden Kennzahlen bzw. ganze Rechenstrukturen “angehängt”.

Die nächste Abbildung zeigt eine vereinfachte Ziel-Mittel-Hierarchie für ein gemeinnütziges Rechenzentrum, z.B. für ein Universitäts-RZ. Die primären und einige sekundären Ziele wurden zum großen Teil aus der Satzung des Leibniz-Rechenzentrums übernommen ([LRZ-WWW]).

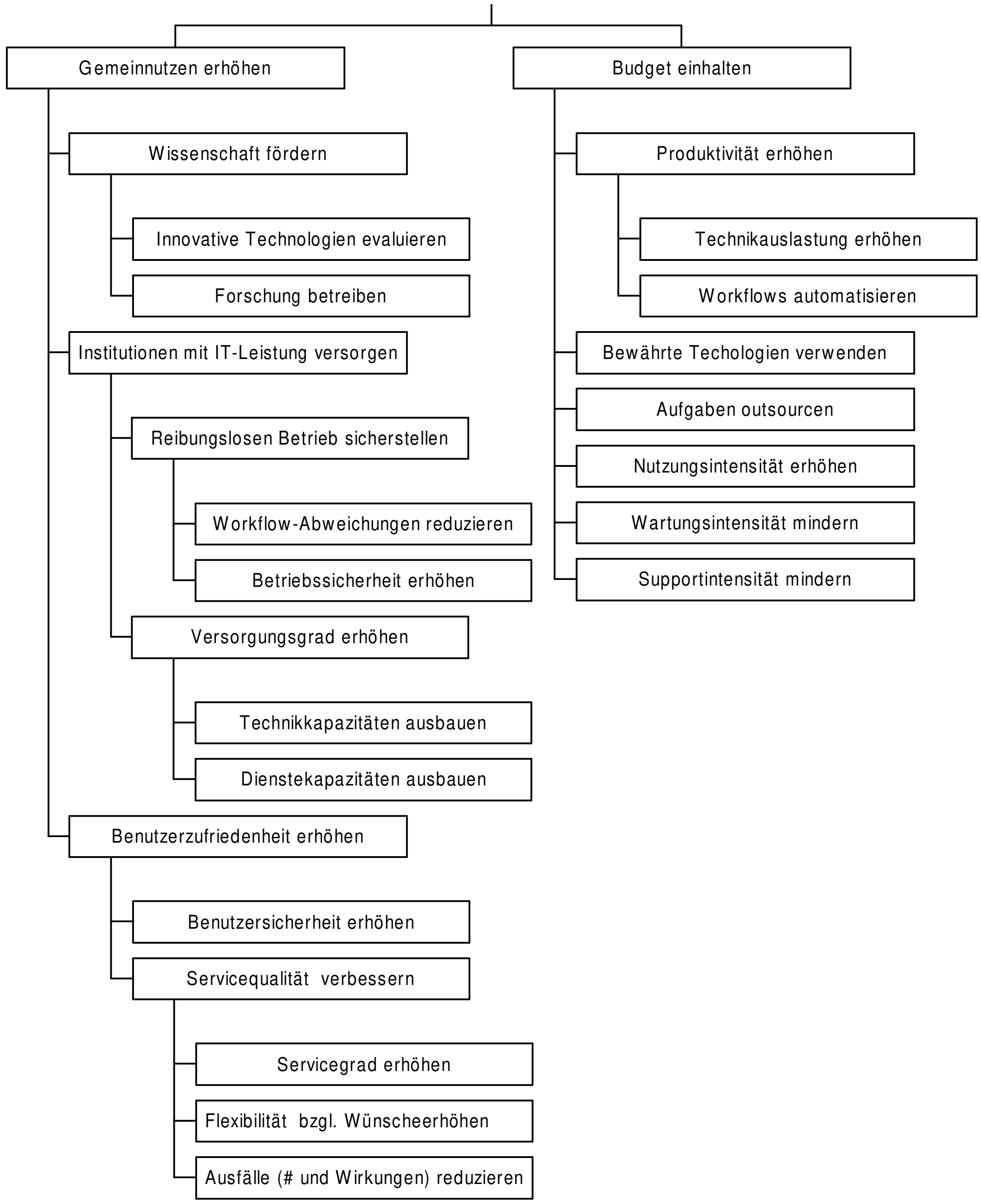


Abbildung 2-11 Ziel-Mittel-Hierarchie für gemeinnützige Rechenzentren

Unter anderem findet man in der LRZ-Satzung folgende Formulierung der Aufgaben und Ziele des Rechenzentrums: "Das Leibniz-Rechenzentrum (LRZ) bietet […] die Möglichkeit, Rechen- und Informationsverarbeitungsaufgaben für wissenschaftliche Forschung und Unterricht durchzuführen. In Zusammenhang damit dient das Leibniz-Rechenzentrum auch der wissenschaftlichen Lehre und Forschung auf dem Gebiet der Informatik. […] Das LRZ betreibt nicht nur das Münchener Hochschulnetz und spezielle Systeme, sondern versteht sich insbesondere als Beratungs- und Kompetenzzentrum für alle Fragestellungen der Informationsverarbeitung."

Wenn man die zwei vorgeschlagenen Hierarchien vergleicht, so fällt einem die Vielzahl von gleichen Zielen und Mitteln auf. So sind praktisch alle technischen Ziele gleichermaßen in der Hierarchie der kommerziellen und der gemeinnützigen Rechenzentren vorhanden. Der Unterschied besteht darin, daß sie als Mittel für verschiedene "nichttechnische" Oberziele verwendet werden. So erhöht ein Universitäts-RZ seinen Innovationsgrad mit dem Ziel der Förderung der Wissenschaft und somit der Erhöhung des Gemeinnutzens. Ein kommerzielles Rechenzentrum macht es dagegen, um seine Marktmacht zu stärken, die Prestige zu verbessern und somit einen Beitrag zur Steigerung des Umsatzes zu leisten.

Aus den Ziel-Mittel-Hierarchien lassen sich konkrete Kennzahlen, eine hierarchische Kennzahlenstruktur (des Typs "Ordnungsstruktur") und Zielsetzungen für Entwicklungen der Kennzahlen ("erhöhen", "reduzieren" u.ä.) ableiten. Wir zeigen in der Abbildung 2-12 einen Vorschlag zur Umsetzung der Zweiges "Gemeinnutzen erhöhen" der Ziel-Mittel-Hierarchie der gemeinnützigen Rechenzentren.

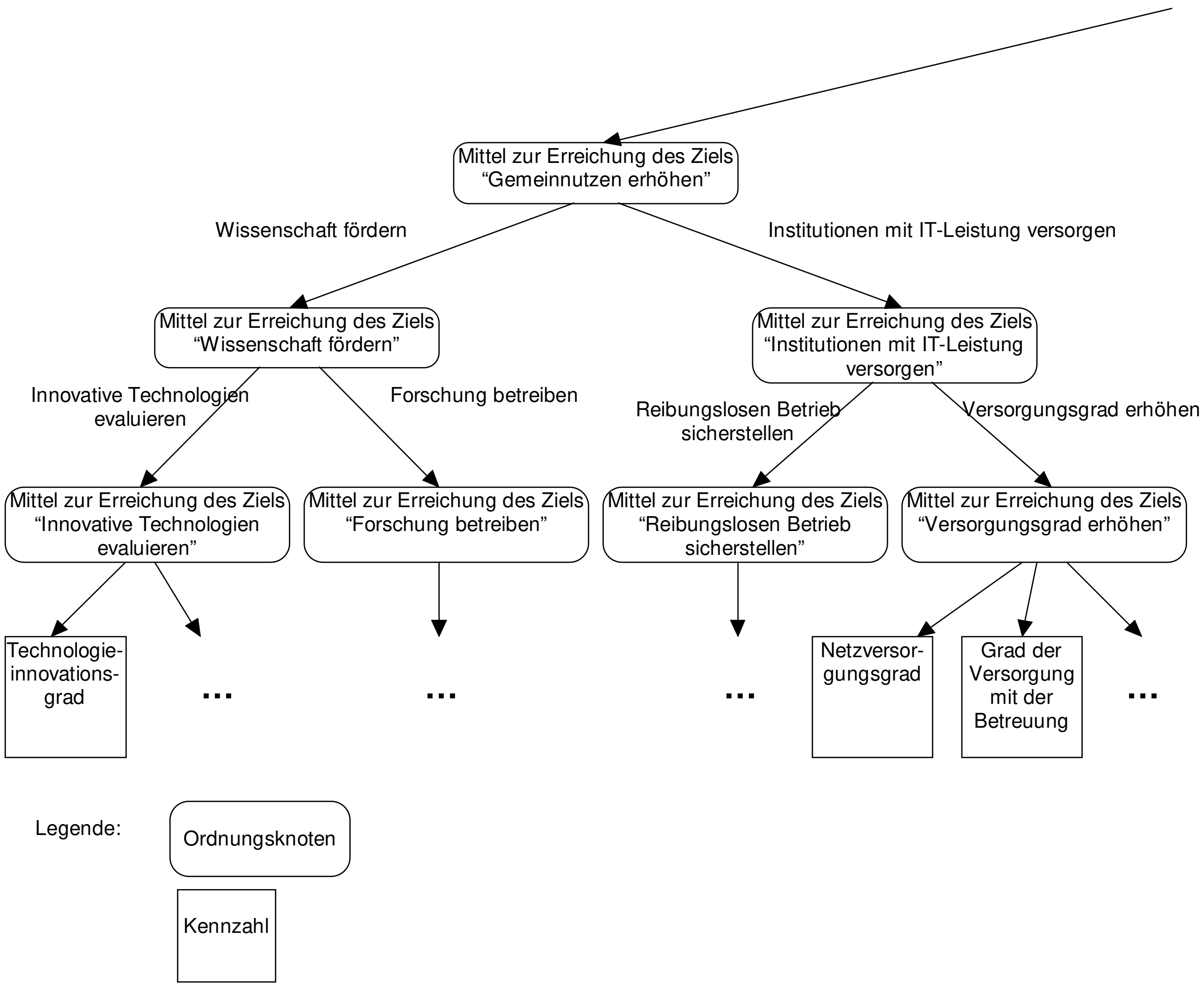


Abbildung 2-12 Ausschnitt aus dem Ziel-Ordnungssystem für gemeinnützige Rechenzentren

Wie man sieht, wurden die Mittel in Kantenmarkierungen umgesetzt (zur Beschreibung der Ordnungsstrukturen siehe Abschnitt 2.1.5). Pro Ziel existiert ein Ordnungsknoten, der die "Mittel"-Kanten bündelt. An der untersten Ebene des Ordnungssystems sind Kennzahlen "angehängt".

2.3 Substruktur "Technische Aspekte des Rechenzentrums"

Am Ende des Abschnitts 2.1.6 haben wir bemerkt, daß für die operative Steuerung des RZ-Betriebs eine Kennzahlenstruktur benötigt wird, in der die technischen Aspekte des Rechenzentrums eingeordnet sind. Unter den technischen Aspekten verstehen wir die technischen Charakteristika der Betriebsmittel, der Workflows und der Dienste/Produkte des Rechenzentrums. Diese Abgrenzung resultiert aus der Diskussion des Begriffs "technische RZ-Kennzahl" im Abschnitt 2.1.1.

In dieser Arbeit wurde eine entsprechende Substruktur entwickelt, die selbst aus drei Teilstrukturen besteht:

- Techn. Charakteristika von techn. Betriebsmitteln,
- Techn. Charakteristika von Workflows und
- Techn. Charakteristika von Diensten.

Das Gesamtbild der Struktur ist in der Abbildung 2-13 zu sehen. Allen drei Teilstrukturen ist gemeinsam, daß sie in der Form eines X-Y-Diagramms aufgebaut sind. Die Y-Achse dient der Darstellung von Managementobjekten, wie z.B. Anwendung A, Plattform B oder Netzsegment C. In der Abbildung werden sie durch Striche auf der Y-Achse vertreten. Die Managementobjekte werden zur besseren Übersicht in Gruppen zusammengefaßt.

Auf der X-Achse werden die jeweiligen technischen Charakteristika eingetragen, wie z.B. die Verfügbarkeit, Anzahl der Ausfälle, Auslastung usw. (Striche auf der X-Achse). Bei der Entwicklung der Teilstrukturen haben wir die bei allen Managementobjekten vorkommenden Eigenschaften zu Gruppen zusammengefaßt. So haben sowohl die Anwendungen als auch die Systemplattformen und Netzsegmente die Charakteristika Ausfälle, Nutzung/Auslastung, Wartungsintensität, Supportintensität u.a. Für alle Typen von Diensten sind die Eigenschaften Servicegrad, Kundenzufriedenheit und Flexibilität von Bedeutung. Jedes RZ-Workflow wird durch seinen Automatisierungsgrad, Abweichungsgrad, Sicherheit u.ä. beschrieben. Die Eigenschaften, die nur für bestimmte Managementobjekte bzw. Objektgruppen sinnvoll sind, werden in besonderen Gruppen “spezifische Charakteristika” zusammengefaßt.

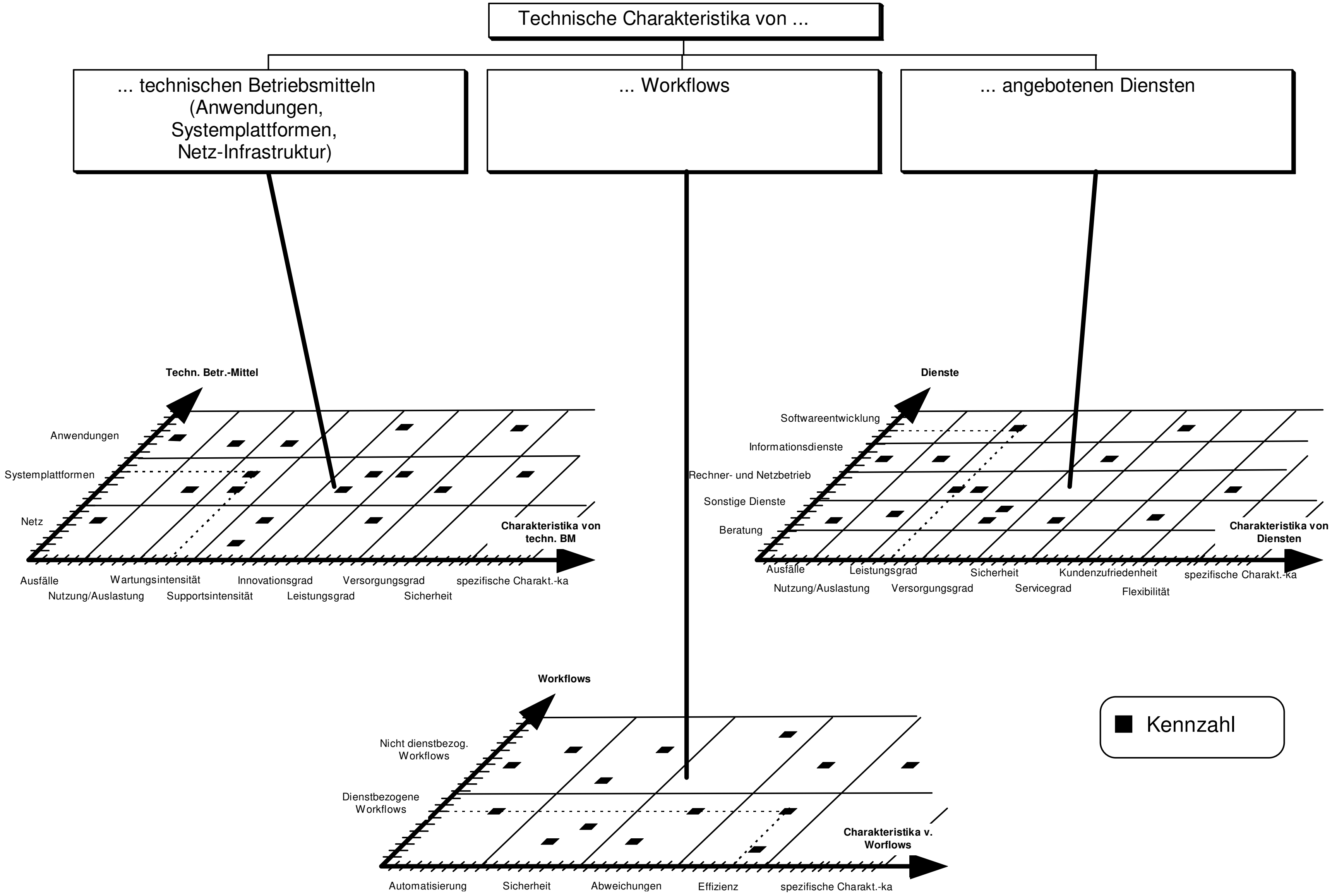


Abbildung 2-13 Technische Aspekte des Rechenzentrums

2.3.1 Teilstruktur "Technische Charakteristika von technischen Betriebsmitteln"

Die Teilstruktur "Technische Charakteristika von technischen Betriebsmitteln" dient der Unterstützung bei der Steuerung von technischen Betriebsmitteln des Rechenzentrums. Als Beispiel für technische Betriebsmittel kann man die Office-Pakete, Datenbanksoftware, Entwicklungswerkzeuge, Workstations, PC's, Server, Drucker, Scanner, Router, Gateways, Access-Server, Kabelleitungen u.a. angeben. Es lassen sich folgende Gruppen von techn. Betriebsmitteln bilden ([Wies96]):

- Anwendungen,
- Systemplattformen und
- Netze.

Für jedes Betriebsmittel sind die Charakteristika(-gruppen):

- Ausfälle(Fehler/Verfügbarkeit),
- Nutzung/Auslastung,
- Leistungsgrad,
- Versorgungsgrad,
- Sicherheit,
- Wartungsintensität,
- Supportintensität und
- Innovationsgrad

sinnvoll (angelehnt an [Wies96]). Spezifische Charakteristika werden in einer besonderen Gruppe zusammengefaßt. In der Tabelle 2-1 findet man unsere Vorschläge für konkrete Kennzahlen, welche die techn. Betriebsmittel beschreiben könnten. Diese Kennzahlen sind für Managementobjekte aus allen drei Gruppen der techn. BM sinnvoll. Die meisten Kennzahlen entnahmen wir aus [Wies96].

Tabelle 2-1 Allgemeine Charakteristika von techn. Betriebsmitteln

Kennzahlen-Gruppe	Kennzahl	Berechnungsvorschrift
Ausfälle	Ausfallhäufigkeit	Anzahl Ausfälle in einer Periode / Dauer der Zeitperiode
	∅Ausfalldauer	Summe Ausfallzeiten /Anzahl Ausfälle
	MTBF[17]	Summe Up-times/Anzahl Ausfälle
	MTTR[18]	Summe Reparaturzeiten/Anzahl Reparaturen
	Verfügbarkeit	(Sollstunden - Ausfallstunden)/Sollstunden
Nutzung/Auslastung	∅ Nutzungsdauer	Summe Nutzungszeiten / Anzahl Logins
	Nutzungshäufigkeit	Anzahl Logins pro Zeitperiode
	Spitzenbedarfszeiten	
	Kapazitätsauslastung	<objektspezifisch>
Sicherheit	Sicherheits-verletzungen	<objektspezifisch>
Wartungsintensität	∅ Wartungsdauer	Summe Wartungszeiten / Anzahl Wartungen
	Wartungshäufigkeit	

[17] MTBF - Mean Time Between Failure
[18] MTTR - Mean Time To Repair

Supportintensität	Supportdauer	
	Supporthäufigkeit	
Innovationsgrad	Alter der eingesetzten Version	
	Integrationsgrad	Anzahl unterstützter Datenaustauschformate u.ä.

2.3.2 Teilstruktur “Technische Charakteristika von Workflows”

Neben den Charakteristika der technischen Betriebsmitteln gehören auch die technischen Eigenschaften von Workflows zu den technischen Aspekten der Rechenzentren. Ein Workflow könnte man folgendermaßen definieren:

Ein Workflow ist der Ablauf einer Diensterbringung oder einer sonstigen Tätigkeit.

Dementsprechend unterscheiden wir zwischen
- dienstbezogenen Workflows und
- sonstigen Workflows.

Zu den dienstbezogenen Workflows in einem Rechenzentrum zählen die:
- Installation, Betreuung und Entwicklung von Anwendersoftware,
- Beratung und Unterstützung,
- Betrieb der Rechner und des Kommunikationsnetzes,
- Sonstige diensterbingende Workflows (z.B. Hilfe bei der Software-, Hardware- und Materialbeschaffung).

Alle diese Workflows resultieren in Diensten, die das Rechenzentrum seinen Kunden anbietet. Im nächsten Unterabschnitt, in dem die Charakteristika der Dienste behandelt werden, findet man deshalb die gleiche Anzahl von identisch benannten Diensten (z.B. die Dienstleistungen “Installation”, “Beratung” oder “Entwicklung”).

Als ein Beispiel für sonstige, nicht dienstbezogene Workflows, kann die Sicherheitsüberwachung des Rechenzentrums genannt werden. Dieses Beispiel verdeutlicht auch den Unterschied zwischen Diensten und Workflows - das Workflow “Sicherheitsüberwachung” stellt keine Dienstleistung der Rechenzentrums dar, sondern dient der Ermöglichung der Erbringung von Diensten.

Bei der Analyse der RZ-Workflows konnten wir vier Charakteristika-Gruppen bilden:
- Automatisierung,
- Sicherheit,
- Abweichungen/Flexibilität und
- Effizienz.

Die Automatisierung eines Workflows besteht in der Ersetzung oder Unterstützung von manuellen Arbeitsschritten durch Informationstechnik. Eine vollständige Automatisierung ist dann erreicht, wenn im Workflow keine manuellen Arbeitsschritte mehr vorkommen. Als Beispiel für ein hundertprozentig automatisiertes Workflow kann man die Anmeldung zu den Praktika an dem Institut für Informatik der Technischen Universität München nennen. Die Studenten füllen, mit Hilfe einer Software, ein Anmeldeformular aus und geben ihre Prioritäten für die einzelnen Praktika an. Am Ende der Anmeldeperiode ordnet das System die Praktikumsplätze den Studenten zu und benachrichtigt sie per Email über die Wahl.

Die Messung der Flexibilität eines Workflows kann z.B. durch die Kennzahlen
- Bearbeitungszeit des Wunsches/Vorschlages und
- Annahmegrad/Abweisungsgrad der Kundenaufräge

erfolgen. Die Sicherheit mißt man z.B. mit dem Anteil der verschlüsselten Datenübermittlungswege. Als eine Kenngröße für die Effizienz könnte die Anzahl der Arbeitsschritte oder die Dauer der Leistungserbringung hergenommen werden.

2.3.3 Teilstruktur "Technische Charakteristika von Diensten"

Die Dienste sind die Schnittstelle des Rechenzentrums gegenüber seinen Kunden. Folgende Dienstleistungen sind üblicherweise im Angebot von Rechenzentren (angelehnt an [LRZ-WWW]):

1) Installation, Betreuung und Entwicklung von Anwendersoftware
2) Beratung und Unterstützung
   - Hotline, allgemeine Benutzerberatung, Fachberatung
   - Geräteanschluß- und Netzberatung
   - Kurse, Veranstaltungen, Publikationen
3) Informationsdienste
   - Benutzerinformationssystem
   - Internet-Dienste, Zugang zu Online-Datenbanken
   - Sonstige Informationsdienste
4) Betrieb der Rechner und des Kommunikationsnetzes
   - Installation, Pflege, Modifikation und Weiterentwicklung der Systemsoftware an den zentralen Anlagen
   - Regelmäßige Dateisicherung an den verschiedenen Rechnern
   - Planung, Aufbau und Betrieb des Kommunikationsnetzes samt der zugehörigen Netzdienste (Nameserver, Mail-Gateways usw.)
   - Installation, Betrieb und Wartung von Datenendgeräten
5) Sonstige Dienste
   - Hilfe bei der Software-, Hardware- und Materialbeschaffung
   - PC-Labor, Workstation-Labor

Diese Dienste werden durch folgende Kennzahlengruppen charakterisiert:
- Ausfälle(Genauigkeit/Fehler/Verfügbarkeit),
- Nutzung/Auslastung,
- Leistungsgrad,
- Versorgungsgrad,
- Sicherheit,
- Servicegrad,
- Kundenzufriedenheit,
- Flexibilität,
- spezifische Charakteristika.

Ähnlich wie bei der Teilstruktur "Charakteristika von technischen Betriebsmitteln" lassen sich allgemeine Kennzahlen-Templates angeben:

Tabelle 2-1 Allgemeine techn. Charakteristika von Diensten

Kennzahlen-Gruppe	Kennzahl
Servicegrad	Termintreue
	Ausführungsverzögerung
	Verweilzeit(Bearbeitungsdauer)
	Erfüllungsgrad der DGV's
Ausfälle	Fehlerhafte Auftragsbeendigungen
	Absolute Fehlerrate
	Genauigkeit
	Relative Fehlerrate
	Zurückweisungsrate
	Verfügbarkeit
Nutzung/Auslastung	Eingegangene Aufträge
	Eingegangene und akzeptierte Aufträge
	Durchschnittliche Auslastung eines Diensterbringers
	Zurückgewiesene Aufträge
	Durchsatz im Intervall
	Abgeschlossene Aufträge
	Fehlerfreie Auftragsbeendigungen
Flexibilität	Bearbeitungszeit des Wunsches/Vorschlages
	Annahmegrad/Abweisungsgrad

2.3.4 Integration der X-Y-Diagramme in die Kennzahlenstruktur

Die X-Y-Diagramme, die wir in der Abbildung 2-13 zur Darstellung der Kennzahlenstrukturen verwendet haben, lassen sich nicht direkt in das Framework integrieren. Unser Framework unterstützt Graphen, jedoch keine X-Y-Diagramme. Deshalb ist es notwendig, diese Diagramme in geeignete Graphenstrukturen umzuformen. In der Abbildung 2-14 sieht man die vorgenommene Umsetzung.

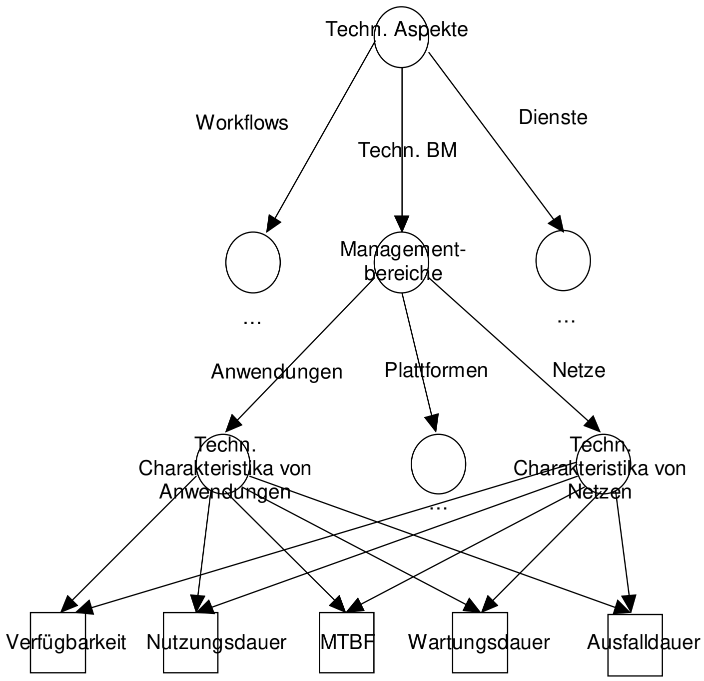


Abbildung 2-14 Ordnungsstruktur "Technische Aspekte von Rechenzentren"

Aus Platzgründen zeigen wir nur den “Ast” der technischen Betriebsmittel. Die beiden anderen Teilstrukturen sind ähnlich abzubilden. Interessant bei der vorgestellten Umformung ist es, daß es nur eine Kennzahl pro allgemeines technisches Charakteristikum gibt. So ist z.B. nur eine Verfügbarkeits-Kennzahl definiert, sie enthält Werte sowohl für Anwendungen, als auch für Netze und Plattformen. Der Anwender kann sowohl die

Verfügbarkeit(“WinWord7”,3/4/1998)

als auch

Verfügbarkeit(“sun1[19]”,3/4/1998), oder
Verfügbarkeit(“br1[20]”,3/4/1998)

über die gleiche Schnittstelle abfragen. Der Preis für diese benutzerfreundliche Schnittstelle ist die größere Komplexität der Berechnungsvorschriften der Kennzahlen.

Natürlich gibt es neben den allgemein anwendbaren Kennzahlen auch spezifische Kenngrößen. Diese werden auf gleiche Weise in die Struktur integriert und dem Ordnungsknoten “spezifische Charakteristika” zugeteilt.

2.4 Substruktur “Benutzergruppen”

Die Nachfrage nach Kennzahlen hängt neben den zu lösenden konkreten betrieblichen Aufgaben auch von der Problemkenntnis, dem Ausbildungsstand und der betrieblichen Funktion des Nachfragers ab. Bei einer groben Einteilung der Benutzer lassen sich folgende drei Gruppen bilden : die Anwender (die Kunden des Rechenzentrums), die Mitarbeiter des RZ’s und das Management des RZ’s.

Ein feinere Differenzierung ist möglich und sinnvoll. So könnte z.B. die Benutzergruppe in “Endanwender” und “DV-Beauftragte” geteilt werden. Die Management-Nutzer sind neben der Leitung des Rechenzentrums auch die Leitung der Abteilungen und Arbeitsgruppen; sie haben ohne Zweifel einen unterschiedlichen Informationsbedarf und sollten deshalb unterschiedliche Menge von Kennzahlen empfangen. Die RZ-Mitarbeiter-Gruppe ist wegen der verschiedenen betrieblichen Funktionen auch nicht einheitlich und kann deshalb in mehrere Gruppen geteilt werden.

In diesem Kapitel bleiben wir bei der Teilung der Benutzer in drei Gruppen “Anwender”, “Mitarbeiter” und “Management”, da die endgültige Definition von Benutzergruppen sowieso nur in einem konkreten Betrieb durchgeführt werden kann. Für den Zweck der Demonstration reicht diese grobe Einteilung vollkommen.

Es ist sinnvoll eine zusätzliche, auf die Benutzergruppen ausgerichtete Substruktur zu definieren (Abbildung 2-15), um die unterschiedlichen Informationsbedürfnisse adäquat zu decken. Diese zusätzliche Struktur ist von dem Typ “Ordnungsstruktur” und hat nur zwei Ebenen von Ordnungsknoten.

[19] sun1 - Den Namen “sun1” verwenden wir in dieser Arbeit zur Bezeichnung einer hypothetischen UNIX-Workstation.
[20] br1 - Mit br1 bezeichnen wir einen hypothetischen Brouter.

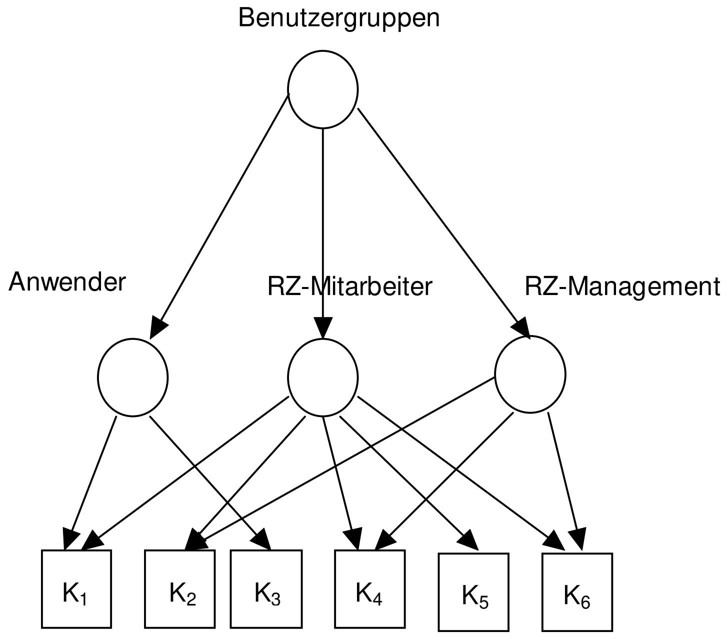


Abbildung 2-15 Substruktur "Benutzergruppen"

Bei der Zuordnung der Kennzahlen zu den Benutzergruppen sollte beachtet werden, daß das Management überwiegend Überblickskennzahlen braucht. Die Granularität der Managementkennzahlen sollte Wochen, Monate oder Quartale betragen - detailliertere Kennzahlen kann die RZ-Leitung wegen der Fülle von Informationen nicht gebrauchen. Nicht nur über die Zeit sollte hier aggregiert werden - es bieten sich auch andere Dimensionen an, wie z.B. die Dimension der Managementobjekte (Aggregation über "alle Systemplattformen", "alle Brouter", über das "Gesamt-RZ").

Die Mitarbeiterkennzahlen haben eine deutlich niedrigere Granularität und beziehen sich auf tägliche, stündliche und noch detailliertere Statistiken von technischen Tatbeständen. Normalerweise werden aber auch alle Überblickskennzahlen gebraucht, denn sie erlauben einen gezielten Einstieg in die Analyse von Detailinformationen.

Kennzahlen, die für die Kunden des Rechenzentrums bestimmt sind, beziehen sich meistens auf die Dienste und Produkte, die das Rechenzentrum anbietet. Das Interna des RZ-Betriebs brauchen die Anwender nicht zu wissen. Die Kunden interessieren Informationen darüber, wie hoch die Qualität der RZ-Dienste ist, wie schnell und flexibel das Rechenzentrum auf ihre Wünsche reagiert usw.

Wegen des zu niedrigen Differenzierungsgrades dieser Kennzahlenstruktur (nur zwei Ordnungsebenen), ist sie allein keine besonders gute Navigationshilfe für die Benutzer des Kennzahlensystems. Sie ist aber auch nicht zum diesen Zweck eingeführt worden, vielmehr soll sie als ein Filter dienen, der in Kombination mit anderen Strukturen die Anzahl der Kennzahlen, mit denen der Benutzer konfrontiert wird, deutlich reduziert. Der Vorteil solch eines Filters ist, daß die Ordnung der Kennzahlen von dem "gefilterten" System übernommen werden kann.

In der Abbildung 2-16 wird auf dem Beispiel einer Ordnungsstruktur gezeigt, wie dieser Filter eingesetzt werden kann. Die Voraussetzung für den Einsatz ist die Zuordnung aller Kennzahlen des Kennzahlensystems zu den Benutzergruppen. Wenn also eine Kennzahlenstruktur vorliegt, die gleichermaßen Management-, Mitarbeiter- und Benutzerkennzahlen enthält, und der Benutzer es wünscht, daß ihm nur die für ihn relevanten Kennzahlen angezeigt werden, so kann die Systemimplementierung die Filtersubstruktur "Benutzergruppen" mit der Ausgangssubstruktur kombinieren und nur diejenigen Kennzahlen anzeigen, die der Gruppe des Benutzers entsprechen. Die Ordnungsstrukturen des Ausgangssystems werden übernommen, somit hat der Benutzer sowohl eine leichte Navigation, als auch eine reduzierte Anzahl von Kennzahlen.

Diese und ähnliche Filterstrukturen sind besonders für sehr umfangreiche Substrukturen wie die “Technischen Aspekte eines Rechenzentrums” wichtig (Abschnitt 2.3), die mehrere Dutzend, wenn nicht Hunderte von Kennzahlen beinhalten können. Ohne die Einbeziehung von Filtern müßten die Entwickler auf Drang der Benutzer mehrere Substrukturen zur Verfügung stellen: eine für das RZ-Management, einige für die Abteilungsleiter, mehrere für die Mitarbeiter mit unterschiedlichen Funktionen. Die Realisierung und Wartung eines solchen Systems wäre wegen der sehr großen Redundanz sehr problematisch. Bei der Existenz des “Benutzergruppen”-Filters braucht man dagegen nur eine allgemeine Substruktur zu definieren, weil das Ergebnis der Kombination dieser Struktur mit dem Filter alle Benutzer des Kennzahlensystems zufrieden stellt.

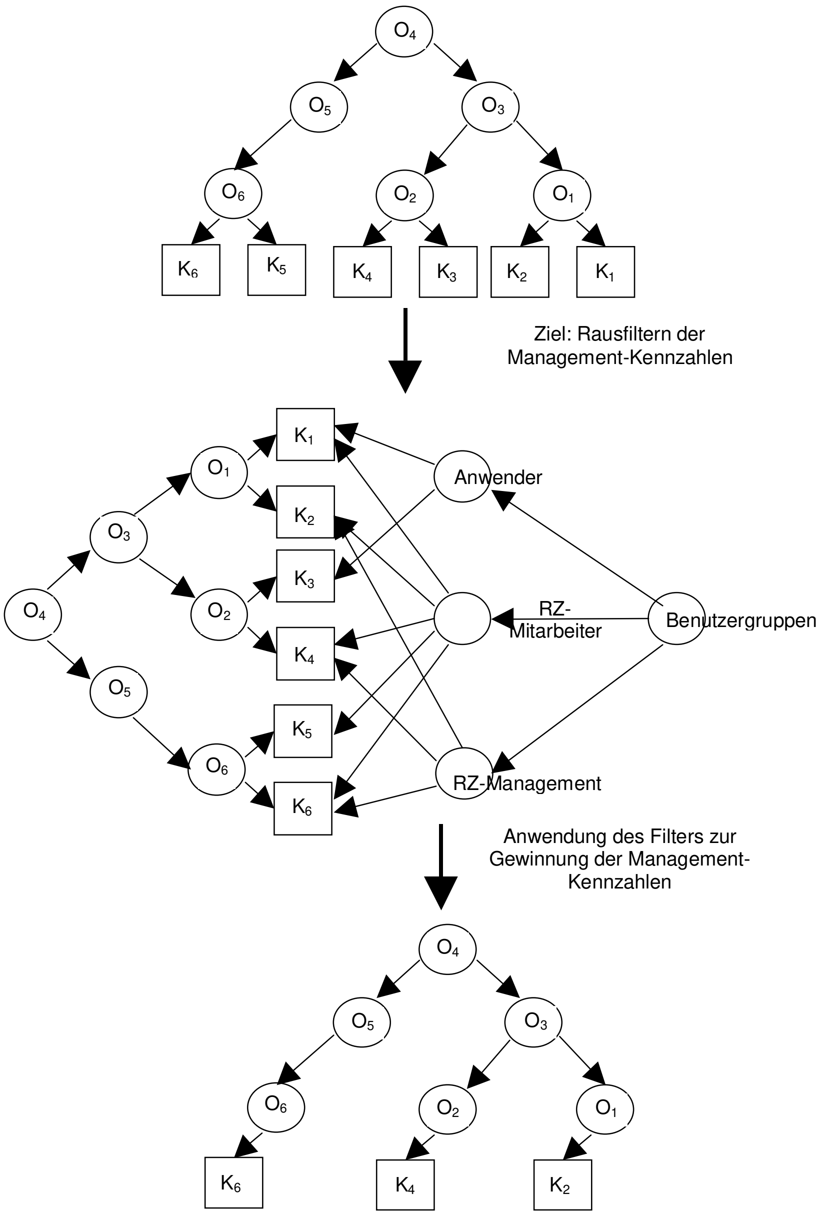


Abbildung 2-16 Anwendung eines Filter-Subsystems

2.5 Zusammenfassung

In diesem Kapitel wurde eine Methode zur Definition von technischen RZ-Kennzahlensystemen vorgestellt. Die Hauptkomponenten der Methode sind die Funktionssichtweise auf die Kennzahlen, der Kennzahlengraph und die Zulassung von mehreren Substrukturen, die auf die einzelnen Aufgabenbereiche (Analyse , Steuerung, Planung, Kontrolle und zwischenbetrieblicher Vergleich) spezialisiert sind.

Zusätzlich zu der Ausarbeitung der theoretischen Grundlagen von Kennzahlensystemen definierten wir in diesem Kapitel drei konkrete Kennzahlenstrukturen. Für die Analyse, Planung und Kontrolle wird eine sog. “Ziel-Mittel-Hierarchie” vorgeschlagen. Zur Unterstützung der Steuerung dient ein Subsystem, das die Charakteristika von technischen Aspekten vereint. Die Definition einer Filter-Substruktur “Benutzergruppen” reduziert die Komplexität bei der Benutzung von umfangreichen Substrukturen des Kennzahlensystems.

3. Eine Realisierungs-Methode für technische RZ-Kennzahlensysteme

Im Kapitel 2 entwickelten wir eine Methode zur Definition von Kennzahlensystemen. In diesem Kapitel diskutieren wir die Realisierung der Kennzahlensysteme, die nach der Methode aus dem Kapitel 2 definiert wurden. Wir beginnen mit den Anforderungen, die an die Realisierung eines Kennzahlensystems gestellt werden (Abschnitt 3.1). Die Diskussion der Lösungsansätze der aufgestellten Anforderungen erfolgt im Abschnitt 3.2. Ein Erkenntnis des Abschnittes 3.2 ist die Feststellung, daß man ein sog. Data-Warehouse-System braucht, um den Anforderungen zu genügen. Im Abschnitt 3.3 erklären wir deshalb die wichtigsten Konzepte dieser Systeme. Basierend auf den ersten drei Abschnitten, stellen wir in den Abschnitten 3.4, 3.5 und 3.6 die Methode vor.

3.1 Anforderungen an die Kennzahlensysteme

Die Kennzahlensysteme sind nichts anderes, als OLAP[21]-Werkzeuge, deshalb werden an sie dieselben Anforderungen gestellt, wie auch an andere entsprechende Informationssysteme. Grob geteilt, besteht ein OLAP-System aus zwei Schichten - der Benutzerschnittstelle und der für die Benutzer nicht “sichtbaren” Basis mit der Datenhaltung, Datenbeschaffung und anderen Funktionen. Die Aufgabe dieser Arbeit war die Entwicklung der Basis eines Kennzahlensystems, nicht aber einer ausgefeilten Benutzerschnittstelle. Deshalb werden wir auch nur solche Anforderungen und Lösungen diskutieren, die eine Relevanz für die Basis haben.

Zu den allgemeinen Anforderungen an die Basis der OLAP-Systeme zählen:
1) Integration der Daten aus verschiedenen operationalen[22] Systemen,
2) Ausgleich der verschiedenen Datenstände in den operationalen Datenquellen,
3) Minimale Beeinflussung der operationalen Systeme,
4) Einhaltung der allgemein akzeptierten Grenzen für die Abfragezeiten von einigen Sekunden,
5) Bereitstellung einer standardisierten Abfrageschnittstelle.

Die Integration von Daten aus verschiedenen Datenquellen ist fast eine Selbstverständlichkeit, ohne diese Funktionalität kann der OLAP-Benutzer keine weitgehenden Analysen durchführen. Wenn ausgehend vom dem OLAP-System die operationalen Daten nicht einheitlich zugreifbar sind, so kann der Benutzer auf das Werkzeug verzichten und bei der in operationalen Systemen vorhandenen Reporting-Funktionalität bleiben.

Das in einem OLAP-System die Daten aus unterschiedlichen Datenquellen stammen können (bzw. müssen), haben wir in obigem Absatz geklärt. Die Datenintegration führt zu einem sehr ernsten Problem der Inkonsistenz des Systems, da in ihm die unterschiedlichen Datenstände der

---

[21] OLAP - OnLine Analytical Processing, eine Bezeichnung für Softwaresysteme zur Unterstützung der Echtzeit-Analyse.
[22] Unter den operationalen Systemen versteht man solche Systeme, die zur Unterstützung des operativen Betriebs eingesetzt werden und deshalb primär auf die Datenerfassung und -verarbeitung spezialisiert sind.

einzelnen Datenquellen in Berührung gebracht werden. So kann z.B. eine Datenquelle nur eine einmonatige Aufbewahrungsfrist für die Daten haben, eine andere Datenquelle enthält die Daten seit der Gründung des Rechenzentrums. Diese unterschiedlichen Datenstände müssen ausgeglichen werden, wenn das System die Grundlage für umfangreiche Analysen sein soll.

Da die OLAP-Systeme die Daten der operationalen Datenquellen benötigen, um sie dem Benutzer zu präsentieren, müssen sie bestimmte Anfragen und Berechnungen in den operationalen Systemen durchführen. Bei der Realisierung der OLAP-Systeme sollten die Entwickler allerdings davon ausgehen, daß die operationalen Systeme bereits an der Grenze ihrer Leistungsfähigkeit liegen. Auch wenn es zum Zeitpunkt der Realisierung des OLAP-Systems nicht ganz stimmt, könnte sich das in einer sehr kurzen Zeit grundlegend ändern. Für die OLAP-Systeme bedeutet das, daß sie nur minimal die operationalen Systeme belasten dürfen, im Falle einer Datenbank etwa wie ein durchschnittlicher User.

Die Benutzer akzeptieren nur solche OLAP-Systeme, die ihre Fragen innerhalb einer angemessenen Zeit beantworten. Das gilt besonders für ad hoc Fragen - der Benutzer wird nur dann mit dem System experimentieren, wenn das schnell geht und die normale Arbeit davon nicht leidet. Aus diesem Grund ist die Performance des Systems eine der wichtigsten Anforderungen, die gestellt werden. Dabei ist nicht die allgemeine Performance gemeint, sondern nur die Performance bei den Abfragen und Berechnungen. Da in dem Kennzahlensystem praktisch keine neuen Daten erfaßt werden, kann die Transaktions-Performance vernachlässigt werden.

Die Basis des OLAP-Systems muß nicht nur perfomant sein, die Daten müssen auch zugreifbar sein. Eine offene Realisierung erlaubt nicht nur der eigenentwickelten Benutzerschnittstelle den Zugriff auf die Daten, sondern auch den Standardwerkzeugen. Die Offenheit ist deshalb wichtig, weil die auf dem Markt befindlichen Präsentations-Tools sehr oft besser sind, als die Eigenentwicklungen. Die Abfrageschnittstelle “SQL” hat sich hier als ein Standard etabliert.

3.2 Lösungsansätze bei der Realisierung

Die aufgestellten Forderungen sind nur mittels einer separaten Datenbank zu realisieren - eine Lösung ohne eigene Datenhaltung würde die Datenquellen unzulässig belasten. Außerdem kann nur mit einer eigenen Datenbank die Ausgleichung der Datenstände vorgenommen werden. Das bedeutet, daß man auf Datenbestände eines gleich großen Zeitraums zugreifen kann, obwohl einige Datenquellen unterschiedliche Aufbewahrungsfristen für ihre Daten haben. Beispiel:

> Falls die Daten über die Ausfälle der Plattformen aus einer Log-Datei des Trap-Dämons genommen werden, so ist die Berechnung der Verfügbarkeitsstatistiken bei einer Lösung ohne einer eigenen Datenhaltung u.U. nur über wenige Tage oder Wochen möglich, da die Log-Dateien aus Platzgründen meistens wöchentlich oder monatlich gelöscht bzw. gekürzt werden. Wenn man jedoch die Log-Datei in einer separaten Datenbank repliziert, und zwar so, daß eine vollständige Historie der Ausfälle entsteht, so können die Verfügbarkeitsstatistiken bis zu jedem gewünschten Datum nachgerechnet werden.

Wenn die Haltung einer Kopie der operationalen Daten in einer separaten Datenbank zwingend erforderlich erscheint, so ist bei der Handhabung der Kennzahlenwerte zwei Alternativen vorhanden. Einerseits können die Kennzahlenwerte, wie im Falle der operationalen Daten, in der Datenbank abgespeichert werden. Die Werte werden nur einmal berechnet und persistent in die Datenbank abgelegt. Alternativ dazu können die Werte immer wieder, wenn der Benutzer sie braucht, berechnet werden, ohne in einer persistenten Datenbankstruktur gespeichert zu werden.

Bei der ersten Lösung haben die Benutzerabfragen die maximale Performance, da keine Berechnungen durchgeführt werden müssen. Als Nachteil ist der erhöhte Speicherplatzverbrauch zu erwähnen. Außerdem ist eine gewisse Redundanz in dem System vorhanden, da die Kennzahlenwerte nichts anderes als Aggregationen der operationalen Daten sind. Die zweite Lösung bietet zwar nicht die maximale Geschwindigkeit, jedoch ist die Belegung des Speicherplatzes minimal. Es gibt auch keine Redundanz in dem System.

Bei der Realisierung eines Kennzahlensystems können beide Lösungen verwendet werden. Eine Kennzahl, die eine Basis für andere Kennzahlen ist[23], sollte in einer persistenten Struktur geführt werden. Die Kennzahlen, auf die nur selten zugegriffen wird, können erst bei Bedarf berechnet werden.

Eine schwierige Frage ist die Wahl der Datenbanktechnologie und eines konkreten Datenbanksystems. Zur Zeit existieren mehrere Ansätze zur Verwaltung und Analyse-Unterstützung von sehr großen Datenmengen. Einerseits gibt es die bewährten relationalen Datenbanksysteme, die sich durch mehrere Jahrzehnte der Optimierung und Weiterentwicklung zur dieser Aufgabe qualifiziert haben. Andererseits existieren proprietäre, aber nicht weniger geeignete sog. multidimensionale Datenbanksysteme. Sie wurden in den letzten Jahren speziell für die Analyse der Unternehmensdaten entwickelt. Sogar die objektorientierten Datenbanksysteme scheinen die Kinderkrankheit der schlechten Performance überwunden zu haben[24] und besitzen darüber hinaus eine bessere und zukunftsträchtigere Technologie.

Wir werden im folgenden von einer relationalen Datenbank ausgehen, da uns einerseits bei der Implementierung des Kennzahlensystems das relationale DBMS Oracle zur Verfügung stand und andererseits die vorgestellten Konzepte leicht auch auf andere DB-Technologien übertragbar sind. Ein Vorteil der relationalen DBMS ist die Verfügbarkeit einer Fülle von Präsentations-Tools, die auf der Grundlage von SQL auf die Daten zugreifen.

Die Überlegungen in diesem und vorherigen Abschnitt können folgendermaßen zusammengefaßt werden. Die Realisierung des Kennzahlensystems soll auf der Grundlage einer separaten Datenbank erfolgen, in der die Inhalte der operationalen Datenquellen gespiegelt werden. Zusätzlich zu der Kopie der operationalen Daten sollen auch einige Kennzahlenwerte in der Datenbank abgelegt werden. Das Datenmodell und die Datenbank-Technologie sind auf die Abfragen zu optimieren. Die Aspekte der Datenerfassung können vernachlässigt werden.

Die obige Zusammenfassung führt zu einem Data Warehouse auf der Basis eines RDBMS. Die Data Warehouses werden als auf Abfragen optimierte, separate Kopien der operationalen Daten definiert, und das ist ja das wichtigste Merkmal der angestrebten Lösung. Aus diesem Grund lehnen wir das Modell unserer Realisierung an das Referenzmodell von Data Warehouses an. In der Abbildung 3-1 wird dieses Referenzmodell präsentiert. Die Erklärung der Konzepte des Data Warehousing und der in der Abbildung verwendeten Begriffe erfolgt im Abschnitt 3.3.

---

[23] Z.B. alle terminale Kennzahlen.

[24] Ein etwas spektakuläres Ergebnis eines neueren Vergleichs : "O_2 (ein bekanntes OO-DBMS, Bem. des Autors) sticht Oracle glatt aus" (Datenbank Fokus, Mai 1997, ObjectFokus-Beilage, S. 6 ff.). Im Test waren relativ große Datenmengen (2,6 GB) im Spiel. Die Performance von O_2 lag nach Angaben des Autors des Berichts um Faktor 2 bis 3 höher, als die des relationalen DBMS Oracle.

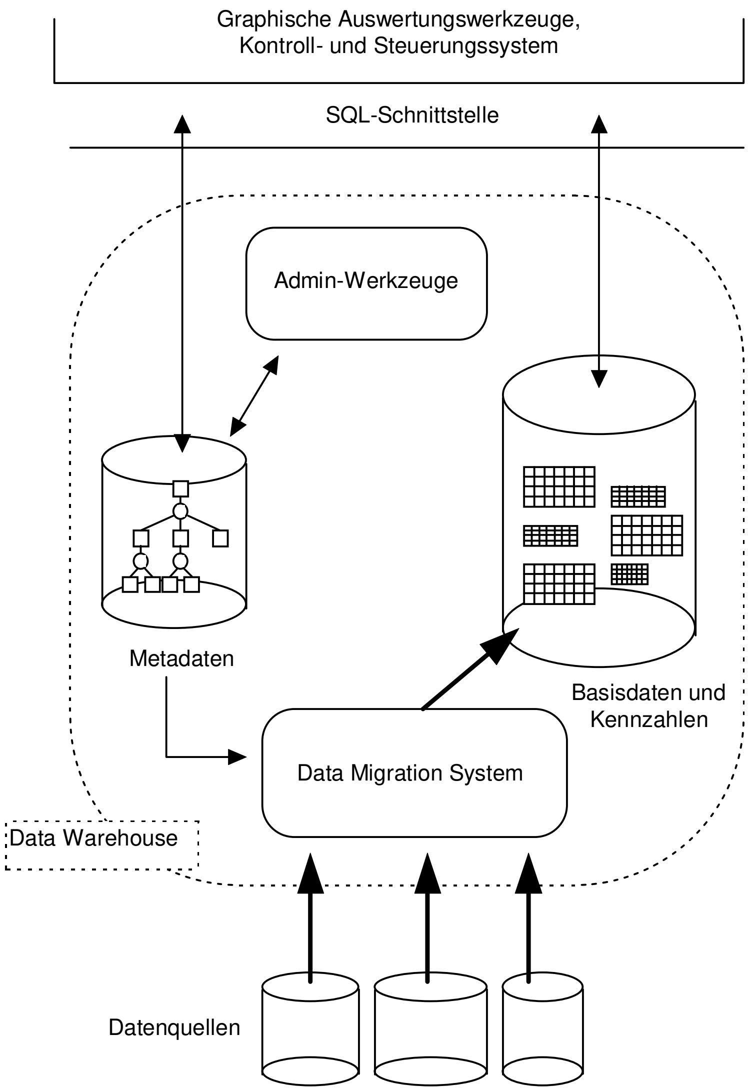


Abbildung 3-1 Referenzmodell der Realisierung

3.3 Konzepte von Data Warehouses

Bevor wir mit der Präsentation der Realisierungsmethode für Kennzahlensysteme beginnen, wollen wir die wichtigsten Konzepte von Data Warehouses in diesem Abschnitt erläutern. Unsere Realisierung eines Kennzahlensystems wird ein ganz normales Data Warehouse sein, deshalb ist es nützlich, die Theorie dieser Systeme kurz zu erklären.

Seit Anfang der 90'er Jahre ist es üblich zwischen zwei Typen von betrieblichen Informationssystemen zu unterscheiden - den sog. operationalen Systemen und den Data Warehouses. Als "operational" bezeichnet man solche Systeme, die hauptsächlich die Aufgabe der Transaktionsverarbeitung, also Datenerfassung, -verarbeitung und -speicherung, haben. Mit ihnen erfolgt die Unterstützung des operativen Betriebs. Für ein Rechenzentrum wären z.B. die Performancemonitore oder Trouble-Ticket-Systeme gute Beispiele für operationale Systeme. Sicherlich besitzen diese Systeme auch eine gewisse Analysefunktionalität (Reports, Listen usw.), jedoch reicht sie für komplexere Analysen nicht aus oder ist nicht perfomant genug. Außerdem sind Analysen, die Daten von mehreren Systemen benötigen, gar nicht oder nur schwer möglich. Dazu kommt noch, daß die Daten flüchtig sind (d.h. nach einer bestimmten Aufbewahrungsfrist gelöscht werden).

Mit der aktuellen Datenbanktechnlogie ist es nicht möglich, in einem System gleichzeitig die Transaktionsverarbeitung und Analysefunktionalität erfolgreich zu vereinigen. Die Anforderungen, die gestellt werden, sind zu unterschiedlich und sogar entgegengesetzt. Außerdem existiert das Problem der unterschiedlichen Datenstände, die nur durch eine separate Datenbank gelöst werden kann. Die Lösung des Analyseproblems heißt seit ca. fünf Jahren "Data Warehouse". Das Konzept des Data Warehousing ist sehr einfach - man versucht nicht mehr alle operationale Systeme mit der Analysefunktionalität aufzurüsten, sondern ein zusätzliches System wird erzeugt. Mehrere operationale Systeme des Unternehmens werden genommen, eine Datentransferschnittstelle gebaut und eine Datenbank mit Daten aus diesen Systemen gefüllt (Data Warehousing ~ Datenlagerung). Falls ein Benutzer die Unternehmensdaten analysieren will, so geht er nicht mehr zu den alten Applikationen, sondern er fragt die neue Datenbank ab. Die Datenerfassung erfolgt wie gewohnt mit den operationalen Systemen.

Dieses Konzept ist keine ideale Lösung. Die ersten Erfahrungen mit diesen Systemen haben gezeigt, daß trotz der Einfachheit der Idee die Kosten für die Implementierung mit fünfstelligen Zahlen gemessen werden und daß es wohl keine große Kostenreduktion möglich ist (die meisten Mittel werden für das Consulting z.B. im Bereich der Datenextraktion ausgegeben und nicht für die Hardware oder Software). Deshalb ist diese Lösung nur für große Unternehmen sinnvoll. Wir sind der Meinung, daß die DBMS-Hersteller schließlich doch versuchen werden, die Transaktionssysteme geeignet zu erweitern, um die künstliche Trennung der Funktionalität zu vermeiden. Für die nächsten Jahre kann man damit aber nicht rechnen.

Wie alle reale Warenlager, benötigt ein Data-Warehouse-System zwei Mechanismen - einen zur Aktualisierung des Bestandes (~ Daten), und einen für die Bedienung der Kunden (~ Benutzer) (siehe Abbildung 3-2).

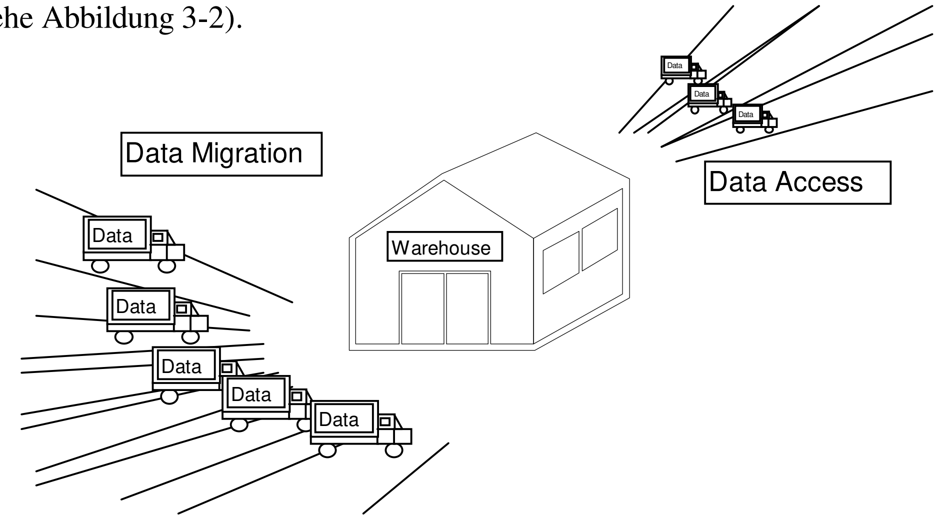


Abbildung 3-2 Zwei Mechanismen von Data Warehouses

Auf Data Access wollen wir in dieser Arbeit nicht eingehen (im Abschnitt 3.1 "Anforderungen an die Kennzahlensystem" erklärten wir den Rahmen dieser Arbeit). Die Konzepte des zweiten Mechanismus (Data Migration), der DW-Datenbank und des sog. Metadaten-Repository werden wir im folgenden kurz erklären.

1) Data-Migration-System

In einem Data-Warehouse-System werden Daten regelmäßig aus operationalen Quellen extrahiert, transformiert und in eine Datenbank geladen. Da die Datenquellen heterogen und verteilt sind, benötigt man zusätzliche Funktionen zur Steuerung und Koordination des Gesamtprozesses. Somit läßt sich die für Migration notwendige Funktionalität in folgende vier Bereiche aufteilen:

- Steuerung und Koordination,
- Extraktion,
- Transformation und
- Laden.

(siehe auch die Abbildung 3-3).

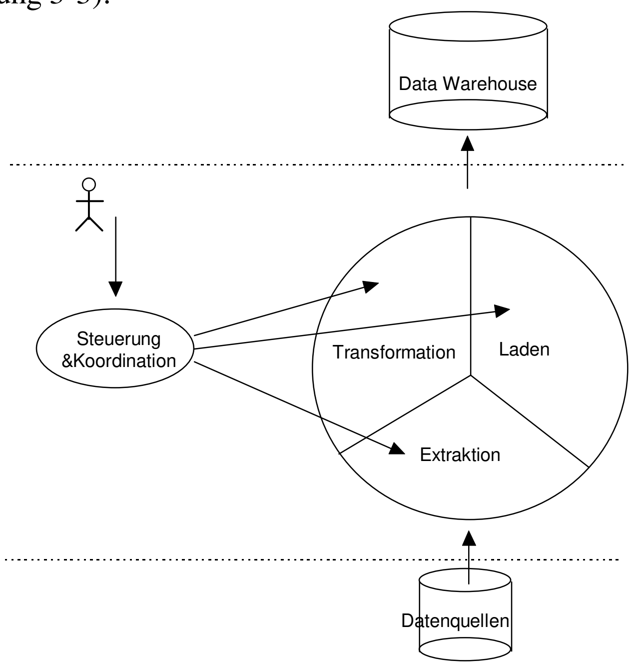


Abbildung 3-3 Funktionale Bereiche von Data Migration

Extraktion

Die Extraktion ist der erste Schritt bei der Migration der operationalen Daten in die DW-Systeme. Ziel ist es, Kopien der operationalen Daten anzufertigen, die durch die Transformations- und Ladefunktionen weiterverarbeitet werden können. Für die konkrete Realisierung der Extraktion ist der Typ der Datenquelle von einer entscheidenden Bedeutung. In Rechenzentren können folgende Typen von Datenquellen vorkommen:

- “Flache” Dateien ohne Strukturinformation, die z.B. Performance-Messungen enthalten,
- Datenbanken von den Managementwerkzeugen, z.B. HP OpenView oder Remedy ARS und
- Netzmanagementagenten auf der Basis von SNMP- oder CMIP-Protokollen.

Transformation

Bevor die extrahierten Daten verwendet werden können, muß man sie geeignet umformen (Transformation). Die Umformung beinhaltet im einzelnen die:

- Validierung der Werte:

(auf dem Beispiel des LRZ)

In der Trouble-Ticket-Datenbank im LRZ existiert pro Trouble-Ticket ein Datensatz. Die Datensätze haben u.a. das Feld “Sachgebiet”. Dieses Feld dient der Zuordnung von Trouble-Tickets zu den einzelnen Sachgebieten, z.B. zu “Netzdienste→Verbindungsproblem” oder “Benutzerinterface PC→Novell-PC”. Wegen interner Schnittstellen-Probleme liefert das Trouble-Ticket-System bei den ersten 4000 Trouble-Tickets für dieses Feld den Wert NULL. In so einem Fall muß das Data-Migration-Systems diese Situation abfangen und z.B. den Wert “unbekannt” in die übernommenen Datensätze schreiben.

- Homogenisierung von Schlüsselwerten:

(auf dem Beispiel des LRZ)

Die Trouble-Ticket-Datensätze im LRZ haben neben dem Feld “Sachgebiet” auch die Felder “TT-Nummer” und “Siehe TT”. Das erste Feld ist der eindeutige Identifikator der Trouble-Tickets, das zweite Feld enthält einen Verweis auf andere Trouble-Tickets, wenn die Probleme in den Tickets in irgendeiner Weise miteinander in Verbindung stehen. Die Werte des Feldes “TT-Nummer” sind immer nach dem Muster “TT9999999” gebildet, z.B. “TT0008214”. Im Feld “Siehe TT” werden aber zur Abkürzung oft nur die letzten signifikanten Ziffern angegeben, z.B. “8214”. Somit gibt es zwei Formate für das Feld “Siehe TT”. Das DM-System muß eine Funktionalität beinhalten, die die unterschiedlichen Kürzel in ein einheitliches Format überführt.

- Integration von Daten aus mehreren Datenquellen, z.B.:

In einem Rechenzentrum können die Performance-Monitore auf mehreren Plattformen installiert sein. Sie protokollieren dann nur den “eigenen” Netzsegment und legen die gesammelten Daten lokal in einer Datei ab. Der Migrator muß diese inhaltlich äquivalenten Daten sammeln und integriert in dem Data Warehouse ablegen.

- Durchführung von mathematischen und betriebswirtschaftlichen Berechnungen,

(auf dem Beispiel des LRZ)

Bei der Extraktion der Datensätze aus der TT-Datenbank wird das Feld “Bearbeitungsdatum” in dem für UNIX üblichen Format “Anzahl Sekunden seit 1.1.1970” exportiert. Falls die DW-Datenbank mit solchen Werten nicht umgehen kann, man aber in dem Warehouse dieses Feld als Datum benutzen will, müssen bei der Migration die Werte z.B. durch Aufrufe der Standard-C-Funktion “strftime“ in das Character-Format transformiert werden. Fast alle Datenbanken unterstützen die Konvertierung von Datumwerten im Character-Format in die internen Datumsformate.

- Aggregation der Daten u.ä.

Als bestes Beispiel für die Aggregation dient die Berechnung der Kennzahlen. Nach der Übernahme der Daten aus den operationalen Datenquellen werden diese zu Kennzahlen verdichtet.

Laden

Das Laden der Daten in die Warehouse-Datenbank ist, verglichen mit der Extraktion und Transformation, erheblich einfacher und leichter. Zum einen hat man hier mit offenen Datenbanksystemen zu tun und nicht mit Dateien oder MIBs. Zum anderen existiert eine große Auswahl von Lade-Tools, die von den Datenbankherstellern angeboten werden.

2) <u>Data-Warehouse-Datenbank</u>

Das Kernelement eines Data Warehouse ist eine eigene zentrale Datenbank. Es liegt nicht nur an den Ressourcenbeschränkungen, die die Verwendung der operativen Datenbanken als

Datenmanager für Data-Warehouse-Anwendungen unmöglich macht. Der spezifische Zugriff auf die Informationen bei der Analyse erfordert auch eine ganz andere Organisation der Datenbank.

In operationalen Systemen ist es z.B. üblich das Datenmodell zu normalisieren, d.h. Redundanzen aus den Daten zu entfernen. Für eine Analyse-Datenbank ist ein vollständig normalisiertes Datenmodell verhängnisvoll, denn es führt dazu, daß eine Mehrzahl von sog. Joins durchgeführt werden müssen. Da es bei der Analyse meistens um viele Tausende, wenn nicht Millionen Datensätze handelt, würden die Joins und andere Operationen zu stundenlangen Berechnungen führen.

Aus diesem Grund speichert man in dem Data Warehouse die Daten meistens denormalisiert, d.h. mit Redundanzen. Um die Datenbankgröße jedoch einigermaßen in Grenzen zu halten , legt man die Daten häufig in einem sog. Sternmodell bzw. Snowflake-Modell ab. Diese Modelle sind ein Kompromiß zwischen der vollständigen Denormalisierung und den üblichen Modellen in den operationalen Systemen.

Ein weiteres Merkmal von DW-Datenstrukturen ist die Existenz von Aggregationen. Im Unterschied zu den Datenmodellen der operationalen Systeme werden sie aber nicht virtuell zur Verfügung gestellt (durch Sichten), sondern physikalisch in der Datenbank abgelegt (in eigenen Tabellen). Die Begründung für diese weitere Redundanz ist wieder die Performance bei den Abfragen. Als Grundlage für die erhoffte Performance-Steigerung dient die Feststellung, daß bei der Analyse fast immer verdichtete Daten benötigt werden und deshalb die häufigsten Datenbankoperationen die Gruppenoperationen sind (Summe, Mittelwert, Abweichung, Maximum, Minimum usw.). Durch Vorberechnung dieser Werte hofft man, den Benutzer schneller “bedienen” zu können.

Die Redundanz, der “Feind” aller operationaler Systeme, hat in den Data Warehouses keine oder eine sehr geringe Bedeutung. In einem Datenerfassungssystem führt die Redundanz im Datenmodell dazu, daß der während einer Transaktion erzeugte, geänderte oder gelöschte Datensatz unter Umständen zur Erzeugung, Änderung oder Löschung von anderen Datensätzen führen kann. Da die Kaskade dieser DB-Operationen u.U. sehr tief sein kann, leidet die Transaktions-Performance darunter sehr, deshalb versucht man durch die Normalisierung des Datenmodells die Redundanz zu minimieren.

In den Data Warehouses fehlen die Transaktionen vollständig, deshalb existiert dort auch das Problem mit der Redundanz nicht. Als einziger Nachteil erweist sich der erhöhte Platzverbrauch, der aber wegen des seit einigen Jahren sehr billigen Sekundärspeichers vernachlässigt werden kann.

3) <u>Metadaten-Repository</u>

Das Metadaten-Repository dient als Speicher für die Informationen über die Datenquellen und das Data Warehouse, enthält also Daten über Daten (deshalb der Namenszusatz “Meta”). Es wird sowohl von dem Data-Migration-System, als auch von den Anwendern des Warehouses benutzt.

Das DM-System bezieht aus dem Repository seine Konfigurationsdaten, wie z.B. den Aktualisierungsplan, die Mappinginformationen , die Transformationsalgorithmen usw. Wir werden unter anderem den Kennzahlengraphen dort abspeichern. Für den Benutzer ist das Metadaten-

Repository eine Quelle für Informationen über die Größe der Tabellen, die besten Analysepfade , die Aktualität der Daten usw.

3.4 Entwicklung der DW-Datenbank

Mit diesem Abschnitt beginnt die Vorstellung der Realisierungsmethode für technische RZ-Kennzahlensysteme. Wir folgen dem Referenzmodell aus der Abbildung 3-1 und fangen mit der Entwicklung der DW-Datenbank an.

Bei der Diskussion der Anforderungen und der Lösungsansätze am Anfang dieses Kapitels stellten wir fest, daß in der DW-Datenbank zumindest die Kopien der operationalen Daten abgelegt werden müssen. Nützlich wäre auch die physikalische Abspeicherung von Werten einiger Basiskennzahlen, um die Bedienung der Benutzerabfragen zu beschleunigen.

In unserer Kennzahlen-Datenbank unterscheiden wir deshalb zwischen drei Typen von Entitäten (Tabellen und Sichten). Diese verschiedene Arten von Entitäten lassen sind durch ein Schichtenmodell veranschaulichen.

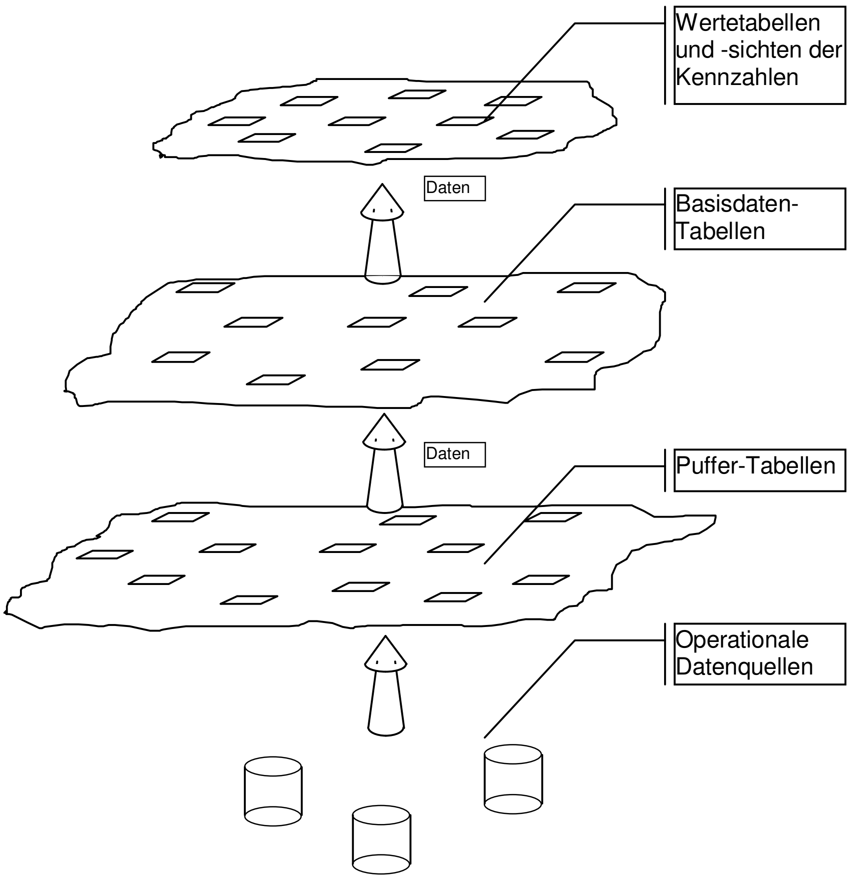


Abbildung 3-4 Schichtenmodell der DW-Entitäten

In den Wertetabellen speichern wir, wie schon aus dem Namen ersichtlich, die Kennzahlenwerte. Diese Tabellen entsprechen den Aggregationen, die in den Data Warehouses sehr üblich sind. Aggregiert werden hier sowohl die Basisdaten, als auch die Werte von anderen Kennzahlen. Möglich sind auch die Wertesichten (Views), in denen die Kennzahlenwerte nicht physikalisch abgespeichert sind, sondern erst zum Zeitpunkt der Abfragen berechnet werden.

Unter dem Begriff “Basisdaten” verstehen wir die Kopien der operationalen Daten, die in dem Data Warehouse abgelegt sind. Die Datenstände in diesen Tabellen sind ausgeglichen, d.h. wurden über einen gleichen Zeitraum gesammelt. In diesem Zusammenhang spricht man auch über die Daten-Historie - in dem Data Warehouse sind Zustände von operationalen Systemen zu finden, die aus den aktuellen Zuständen einfach nicht rekonstruierbar wären.

Die Inhalte der Basisdatentabellen basieren nicht direkt auf den operationalen Datenquellen, es existiert eine Zwischenschicht aus Puffertabellen.

Die Erklärung des Drei-Schichten-Modells richten wir nach dem Datenfluß aus und beginnen deshalb mit den Puffertabellen.

1) Puffertabellen

Die Puffertabellen werden aus Implementierungsgründen benötigt. Sie dienen nur dazu, um so schnell wie möglich die operationalen Daten in das Warehouse zu schaffen. Dadurch hoffen wir bereits in der frühen Phase der Datenübernahme auf die Mechanismen des DBMS zugreifen zu können, um die nötigen Transformationen der Daten zu vollziehen. Ihr Datenmodell ist sehr an das Format der Datenschnittstellen der Datenquellen angelehnt. Wegen der Nähe des Pufferschicht-Datenmodells an den Datenquellen-Schnittstellen, ist dieses nicht genügend abstrakt und beinhaltet viele Redundanzen. Das Datenmodell der Basisdaten ist dagegen deutlich abstrakter, die Redundanzen sind auf das Minimum reduziert (→Snowflake-Modell). Eine weitere Besonderheit der Puffer liegt darin, daß ihre Inhalte nach der erfolgreichen Datenübernahme in die Basisdatentabellen vollständig gelöscht werden, so daß sie vor jedem Aktualisierungszyklus leer sind.

2) Basisdaten

Wir schlagen vor, für alle verfügbaren Datenquellen eigene Relationen (eine oder mehrere) im Data Warehouse einzurichten, in denen ihre Daten dupliziert werden. Wir nennen diese Relationen “Basisdatentabellen” und kürzen sie mit “BD-Tabellen” ab.

Weshalb die Abspeicherung der operationalen Daten innerhalb des Kennzahlensystems notwendig ist, haben wir am Anfang des Kapitels ausführlich erklärt. Die Frage, wieso möglichst viele der verfügbaren Datenquellen im Warehouse abgespeichert werden sollten, auch wenn man noch keine Kennzahlen auf ihrer Basis definieren will, läßt sich so beantworten. Durch die Einbeziehung von allen verfügbaren Datenquellen in das Data Warehousing wird die Flexibilität des Systems bzgl. der neuen Anforderungen und Erkenntnisse erhöht. So kann die Definition neuer Kennzahlen sehr schnell und leicht realisiert werden. Außerdem können empirisch-statistische und Data-Mining-[25]Untersuchungen durchgeführt werden, die zu neuen Kennzahlenstrukturen und Kennzahlen führen könnten.

3) Wertetabellen und -sichten der Kennzahlen

Wertetabellen und -sichten bilden die Abfrageschnittstelle des Kennzahlensystems gegenüber den Präsentations-Tools oder dem Benutzer. Die Auswahl der richtigen Realisierungsform

[25] Data Mining wird definiert als die Suche in Datenbanken nach implizit vorhandenem, nicht trivialem und nützlichem Wissen. Ein Data-Mining-Prozeß analysiert den Datenbestand und findet Klassen von gleichartigen Datensätzen und Regeln, die für diese Klassen charakteristisch sind. Die Regeln können dann als Beschreibungen des Datenbestands verwendet werden. So können z.B. funktionale Zusammenhänge zwischen Kennzahlen festgestellt werden.

(Sicht oder Tabelle) für eine Kennzahl hat einige Feinheiten, die die Entwickler der Kennzahlensysteme unbedingt berücksichtigen müssen.

Erstens, in einer View ist es unmöglich, die Daten physikalisch abzulegen. Somit fällt diese Realisierungsmöglichkeit für solche Kennzahlen weg, die z.B. zwischen Soll- und Istwerten unterscheiden müssen. Die Sollwerte werden nämlich von dem Benutzer vorgegeben und können deshalb nicht auf der Basis der operationalen Daten berechnet werden.

Zweitens, sollten diejenigen Kennzahlen als Tabellen implementiert werden, auf die man oft zugreift, oder die eine Basis für andere Kennzahlen bilden. Kennzahlen, die kurzfristig einen sehr konkreten Informationsbedarf befriedigen müssen oder nur selten gebraucht werden, sollten lieber als Sichten definiert werden - das spart Speicherplatz in dem Warehouse und beschleunigt die Aktualisierung.

Für jede Kennzahl wird im Data Warehouse entsprechend ihrer Funktionalität

$$K: \langle Parametertyp_1 \rangle \times \langle Parametertyp_2 \rangle \times \ldots \times \langle Parametertyp_n \rangle \rightarrow \langle Wertetyp \rangle$$

eine Tabelle oder Sicht für ihre Werte erzeugt. Zur Vereinheitlichung der Namengebung wird vorgeschlagen, die Tabellen bzw. Sichten nach dem Muster **WERT_**<Kennzahl-Id> zu benennen, z.B. WERT_K1, WERT_K2 usw. Die Funktionalität der Kennzahl wird folgendermaßen in die DDL[26]-Anweisungen umgesetzt:

```
CREATE TABLE WERT_<Kennzahl-Id>(
    WERT_ID              NUMBER[27],          -- Eindeutige Wert-Id, Primärschlüssel
    <Parametername_1>    <Parametertyp_1>,    -- Erster Parameter
    …,
    <Parametername_n>    <Parametertyp_n>,    -- letzter Parameter
    WERT                 <Wertetyp>)          -- Wert der Kennzahlenfunktion
```

bzw.

```
CREATE VIEW
WERT_<Kennzahl-Id>(<Parametername_1>,…,<Parametername_n>,WERT) AS SELECT ….;
```

[26] DDL - Data Definition Language. DDL ist eine Untermenge von SQL, der Structured Query Language, die eine Standard-Sprache der relationalen DBMS ist.
[27] In diesem und folgenden Abschnitten werden die ANSI SQL-2 Bezeichnungen für Datentypen verwendet

Beispiel:

Sei die Kennzahl K1 "Sachgebiete stündlich" (definiert im Anhang A) mit der Funktionalität

fct K1 = (Sachgebiet_1: **string**, Sachgebiet_2: **string** , Sachgebiet_3: **string**, Stunde : **date**) **number**

gegeben. Diese Kennzahl beantwortet die Frage: "Wieviele Trouble-Tickets eines bestimmten Sachgebiets wurden in einer bestimmten Stunde erfaßt ?". Für diese Kennzahl wird die folgende DDL-Anweisung generiert:

```
CREATE TABLE WERT_K1(
    WERT_ID         NUMBER           PRIMARY KEY,
    SACHGEBIET1     VARCHAR2(64)     NOT NULL,
    SACHGEBIET2     VARCHAR2(64),
    SACHGEBIET3     VARCHAR2(64),
    STUNDE          DATE             NOT NULL,
    WERT            NUMBER           NOT NULL,
    UNIQUE (SACHGEBIET1,SACHGEBIET2,SACHGEBIET3,STUNDE)
);
```

Die Kennzahl K4 "Spitzenbedarfszeiten" könnte dagegen als eine View auf die Tabelle WERT_K1 definiert werden:

```
CREATE VIEW WERT_K4 (STUNDE, MONAT, WERT) AS
SELECT
    TRUNC(24*(STUNDE - TRUNC(STUNDE,'DD'))),
    TRUNC(STUNDE,'MM'),
    SUM(WERT)*0.05
FROM WERT_K1
GROUP BY TRUNC(STUNDE,'MM'),TRUNC(24*(STUNDE - TRUNC(STUNDE,'DD')));
```

Die Kennzahl K4 beantwortet die Frage : "Wieviele Trouble-Tickets werden in einer bestimmten Stunde des Arbeitstages durchschnittlich erfaßt?". Der Durchschnitt wird über alle Tickets eines Monats gebildet.

3.5 Entwicklung des Metadaten-Repository

Im Abschnitt 3.4 gingen wir auf die Entwicklung der aus der Sicht des Benutzers wichtigsten Komponente des Data-Warehouse-Systems - der DW-Datenbank - ein. Für die Entwickler ist jedoch ein ganz anderes Teil des DW-Systems von entscheidender Bedeutung - gemeint ist das Data-Migration-System. Die Berichte von abgeschlossenen Data-Warehouse-Projekten deuten darauf hin, daß mehr als die Hälfte[28] der Ressourcen in die Migration der Daten aus den operationalen Quellen in die DW-Datenbank investiert werden müssen.

Bei der Realisierung des DM-Systems kann man entweder die Entwicklung vollständig auf die augenblickliche Situation ausrichten und die Migrations-Parameter (z.B. den Aktualisierungsplan oder die Zugriffsmethoden auf die operationalen Datenquellen) in das System hart kodieren, oder man kann die Migrations-Parameter in einer Datenstruktur ablegen und das DM-System allgemein gültig entwickeln. Der Vorteil der zweiten Lösung ist die größere Flexibilität, und sie (die Flexibilität) wird in den DW-Systemen dringend gebraucht. Aus diesem Grund

[28] William Inmon, der "Vater" des Data-Warehouse-Konzeptes, spricht sogar von über 80%.

ist sowohl in fast allen kommerziell angebotenen Data Warehouses, als auch in vielen Eigenentwicklungen, die Funktionalität der Parameter-Speicherung und der eigentlichen Datenmigration getrennt realisiert. Als ein zusätzlicher Vorteil der Funktionalitäts-Trennung erweist sich die Möglichkeit der Benutzung des Metadaten-Repository durch die Anwender des Warehouses, die dort z.B. die Aktualität der Daten oder die günstigsten Analysepfade entnehmen können.

Wenn bei der Wahl der richtigen DB-Technologie für die Speicherung der Kennzahlenwerte und der Basisdaten besonders viel Wert auf die Performance gelegt werden muß, so werden für das Metadaten-Repository keine besonders perfomanten Datenbanken gebraucht. Die Mengen von Metadaten, die hier anfallen, sind nur ein Bruchteil der "echten" Daten. Das Hauptkriterium bei der DBMS-Auswahl für das Repository ist die Unterstützung des besonderen Datenmodells der Metadaten.

Die Vielfalt der zu Metadaten gehörenden Entitäten (bzw. Klassen) ist deutlich größer als die Vielfalt bei den Kennzahlen und Basisdaten. Beispielsweise lassen sich folgende Klassen identifizieren : Kennzahlengraph, Datengraph, Knoten, Kante, Kennzahl, Vorschrift, Datenquelle, Basisdatentabelle, Differenzierungskriterium, Wert des Differenzierungskriteriums, Substruktur, Heuristik, emp.-stat. Zusammenhang, Aktualisierungsplan, Benutzer usw. Es liegt nahe einen objektorientierten Ansatz zu wählen, um das Repository zu entwickeln.

Die meisten der genannten Metadaten-Klassen haben wir bereits im Kapitel 2 angesprochen. Uns bleibt nur noch den sog. "Datengraphen" einzuführen, der die Zusammenhänge zwischen den Kennzahlen, Basisdatentabellen und Datenquellen beinhaltet.

Der Datengraph ist eine zusätzliche Struktur, welche die Abhängigkeiten zwischen den Kennzahlen, Basisdatentabellen und den Datenquellen speichert. In dem Datengraphen ist für jede terminale Kennzahl vermerkt, welche Basisdatentabellen für die Berechnung der Kennzahlenwerte benötigt werden. Außerdem sind für jede Basisdatentabelle die Datenquellen aufgelistet, die die operationalen Daten liefern.

Dieser Graph wird dazu gebraucht, um bei der Aktualisierung eines Teiles der Kennzahlen die minimale Menge der Datenquellen bestimmen zu können, die für die Aktualisierung notwendig sind. Es ist dann möglich, ad hoc die aktuellen Werte von wenigen Kennzahlen zu ermitteln, ohne das gesamte Warehouse aktualisieren zu müssen. Besonders in großen Kennzahlensystemen mit Hunderten von Kennzahlen, sehr umfangreichen Kennzahlenstrukturen und Dutzenden von Datenquellen ist die Möglichkeit der partiellen Aktualisierung der Kennzahlenwerte sehr hilfreich.

Wir zeigen den allgemeinen Fall eines Datengraphen in der Abbildung 3-5.

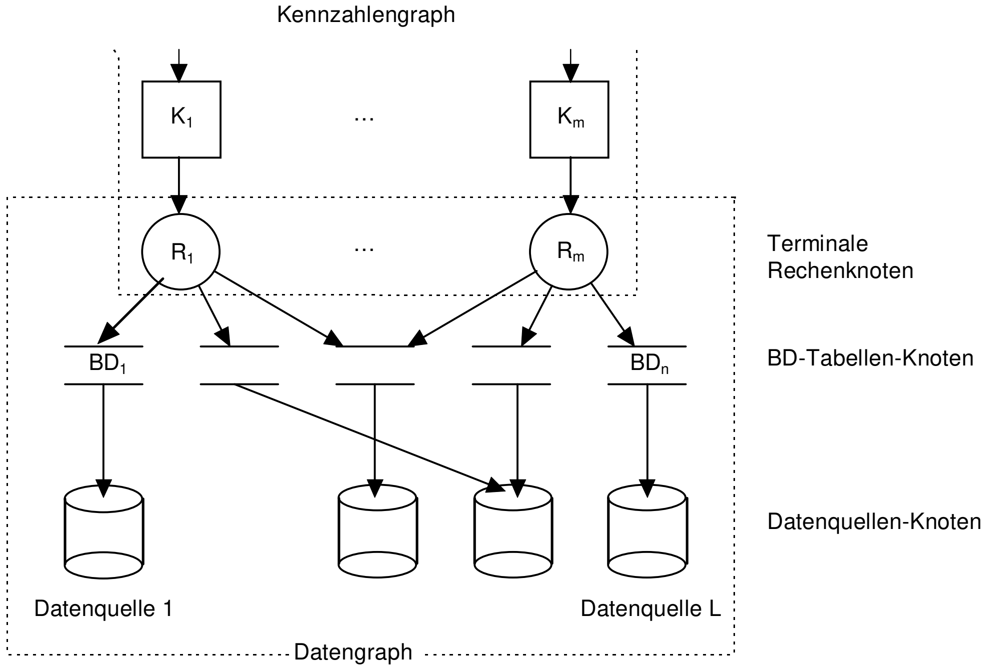


Abbildung 3-5 Datengraph

Der Datengraph überlappt sich mit dem Kennzahlengraphen - beide Graphen enthalten die terminalen Rechenknoten. In ihm gibt es zusätzlich zu den Rechenknoten noch weitere zwei Knotentypen - die Knoten, die die Tabellen mit den Basisdaten repräsentieren, und die Datenquellen-Knoten. Wir definieren den Datengraphen folgendermaßen:

Ein Datengraph ist eine Struktur zur Repräsentation von Abhängigkeiten zwischen Kennzahlen und Datenquellen. Er wird mathematisch durch einen gerichteten Graphen modelliert.

3.6 Entwicklung des Data-Migration-Systems

Im letzten Abschnitt begannen wir die Vorstellung des Metadaten-Repository mit der Bemerkung, daß die aus der Entwicklersicht wichtigste Komponente eines Data Warehouses das Data-Migration-System ist. In diesem Abschnitt führen wir die Analyse des Data-Migration-Systems durch. Entsprechend dem Referenzmodell aus der Abbildung 3-1 sorgt das Data-Migration-System dafür, daß die Daten aus den operationalen Datenquellen in regelmäßigen Abschnitten in die DW-Datenbank gelangen und daraus die neuen Kennzahlenwerte berechnet werden.

Als Analysemethode wählten wir die "Structured Analysis" (kurz : SA) von Tom de Marco ([Marc78]). Der Grund für die Auswahl der Methode war der, daß der Prozeß der Initialisierung und Aktualisierung der DW-Datenbank (d.h. der Prozeß der Data Migration) einem Datenfluß gleicht, der bei den operationalen Datenquellen beginnt, durch Extraktions-, Transformations- und Lade-Prozesse umgeformt wird und in der DW-Datenbank mündet (vgl. Abbildung 3-3 Funktionale Bereiche von Data Migration). Die Strukturierte Analyse ist in solchen Fällen besonders geeignet, da die Hauptkomponenten ihrer Modelle die Datenflüsse (Data Flows) und Prozesse (Processes) sind.

Die Analyse nach SA besteht darin, daß das Problem mit mehreren sog. Data Flow Diagrams (DFD's) beschrieben wird. Die Diagramme bestehen aus vier Hauptelementen (siehe auch Abbildung 3-6):

- Datenflüssen (im Original *data flow* genannt), repräsentiert durch markierte Pfeile,
- Prozessen (*process*), repräsentiert durch Kreise (auch "Bubbles" genannt),
- Dateien (*file*), repräsentiert durch zwei parallele Linien, und
- externen, dem System nicht angehörenden Datenquellen und -senken (*terminator*), repräsentiert durch Quadrate.

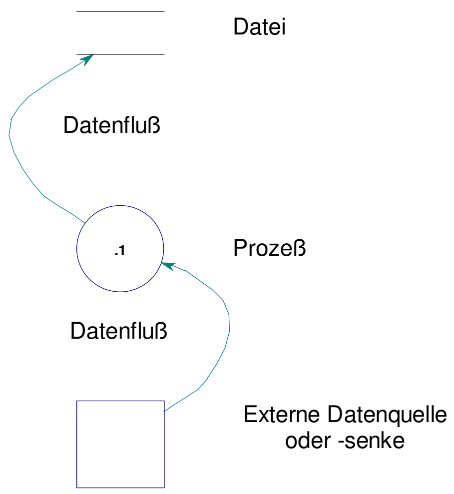


Abbildung 3-6 Elemente der Data Flow Diagramme (DFD)

Zwischen Prozessen, Dateien und externen Datenquellen (und -senken) können Daten fließen, wobei zwischen zwei Dateien immer ein Prozeß sein muß. Die Diagramme bilden eine Hierarchie, d.h. die nichtterminalen Prozesse in einem Diagramm werden in weiteren Diagrammen näher spezifiziert. Dabei muß gelten, daß die einfließenden und rausfließenden Ströme in beiden Diagrammen vorhanden sein müssen. Das oberste Diagramm der Hierarchie heißt das Kontext-Diagramm und sollte alle externen Datenquellen und -senken zeigen. Terminale Prozesse bilden die Blätter des Hierarchiebaums und werden durch sog. Mini Specs[29] weiter spezifiziert.

3.6.1 Kontext des Systems

Gegeben sind mehrere operationale Datenquellen, die Managementinformationen enthalten, und eine Warehouse-Datenbank, die eine Basis für die Berechnung von Kennzahlenwerten werden soll. Das Ziel ist es, ein System zu konstruieren, das diese Datenbank initialisiert und nach einem bestimmten Zeitplan aktualisiert. Zusätzlich zu der planmäßigen Aktualisierung ist auch eine ad hoc Aktualisierung, initiiert durch den Benutzer, zu unterstützen.

Wir fangen mit dem Kontext-Diagramm des Aktualisierungssystems an (Abbildung 3-7). Wie man sieht, bezieht das System Daten aus mehreren (L) operationalen Datenquellen (Data Flows "Inhalt_Operationale_Datenquelle_*"). Der Benutzer oder ein Aktualisierungsdämon beauftragen das System mit der Aktualisierung von bestimmten Kennzahlen ("Liste_der_zu_aktualisierenden_Kennzahlen") und erhalten eine Liste von tatsächlich aktuali-

[29] Mini Spec - Abkürzung für Mini Specification. Textuelle Spezifikation eines Prozesses z.B. in einer abstrakten prozeduralen Sprache.

sierter Kennzahlen zurück. Das System aktualisiert intern die Warehouse-Datenbank, auf die der Benutzer z.B. über ein Standardabfragetool des DBMS-Herstellers zugreifen kann.

Im Kontext-Diagramm sind keine Datenflüsse mit Kennzahlenwerten zu finden - die Bedienung der Abfragen ist nicht die Aufgabe des Aktualisierungssystems. Die Initiierung des Aktualisierungsprozesses wurde auch aus dem System ausgelagert, denn diese Funktionalität wird in verschiedenen Umgebungen unterschiedlich realisiert. Die einzige Forderung an die Prozesse, die die Aktualisierung initiieren, ist die Einhaltung der Schnittstelle zum Aktualisierungsprozeß, definiert durch den Input-Datenstrom “Liste_der_zu_aktualisierenden_Kennzahlen” und Output-Datenstrom “Liste_der_aktualisierten_Kennzahlen”.

Bei der Vorstellung des Datengraphen im vorigen Abschnitt haben wir darauf hingewiesen, daß die partielle Aktualisierung des Data Warehouses (Basisdaten und Kennzahlenwerte) große Vorteile bei umfangreichen Kennzahlensystemen bringt. Unter Umständen kann die Datenbank unternehmensweite Basisdaten und Werte von Hunderten Kennzahlen enthalten und dementsprechend groß sein. Wenn in so einer Umgebung der Benutzer ad hoc die aktuellen Werte einiger Kennzahlen haben will (z.B. um kurzfristig zu entscheiden, ob eine Plattform neu gebootet werden soll), so muß er bei einer Lösung, die keine Auswahl von zu aktualisierenden Kennzahlen erlaubt, mehrere Stunden warten, bis die Gesamtaktualisierung abgeschlossen ist. Die Aufgabe der Unterstützung der operativen Steuerung wäre in diesem Fall sehr schlecht erfüllt.

Aus diesem Grund haben wir in unsere Systemrealisierung die Möglichkeit eingebaut, nur einige ausgewählte Kennzahlen aktualisieren zu können, und das mit einem Aufwand, der gering wie nur möglich ist. Der Preis für die höhere Flexibilität ist die schwierigere Implementierung, da die Komplexität der Steuerung und Koordination höher wird. U.a. muß die minimale Menge der Datenquellen bestimmt werden, auf deren Basis die Kennzahlenwerte vollständig berechnet werden können. Weitere Analysen auf der Grundlage des Kennzahlen- und Datengraphen sind auch notwendig.

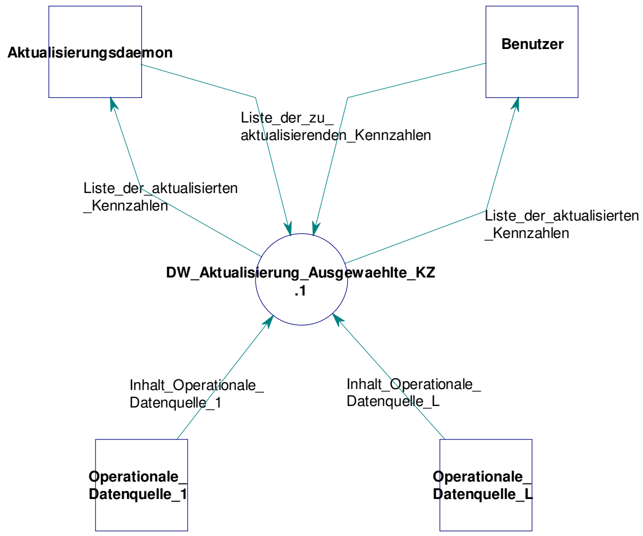


Abbildung 3-7 Kontext-DFD

3.6.2 Aktualisierung von Kennzahlen : Überblick

In diesem Abschnitt begeben wir uns auf die erste Hierarchieebene unseres Aktualisierungssystems (das Kontext-Diagramm war die 0-te Ebene) und behandeln die Aktualisierung von ausgewählten Kennzahlen. Die Abbildung 3-8 zeigt das Data Flow Diagramm des Prozesses

“DW_Aktualisierung_Ausgewaehlte_KZ”. Dieser Vorgang ist der Hauptprozeß des Data-Migration-Systems. Das Ziel ist es, die Aktualisierung mit einem minimalen Aufwand durchzuführen, d.h. nur diejenigen Datenquellen abzufragen und nur solche Basisdatentabellen zu aktualisieren, die wirklich für die Erneuerung der Werte von ausgewählten Kennzahlen benötigt werden.

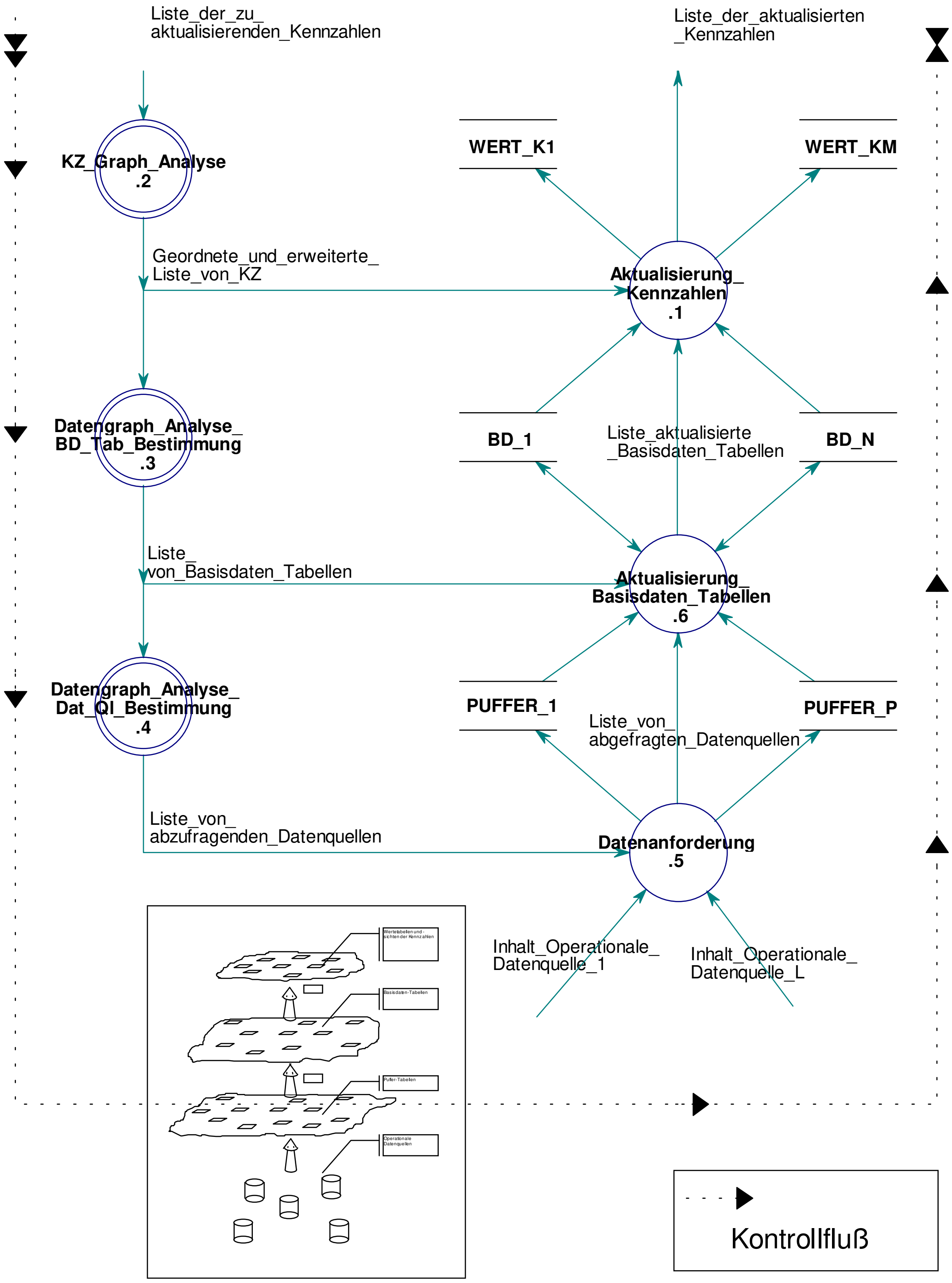


Abbildung 3-8 DFD “DW_Aktualisierung_Ausgewaehlte_KZ”

Auf dem Weg von der Liste der zu aktualisierenden Kennzahlen bis zu der Auswahl von abzufragenden Datenquellen müssen weitgehende Analysen des Kennzahlen- und Datengraphen durchgeführt werden. Wir haben diese Aufgabe in drei folgende Unteraufgaben geteilt:

- Bestimmung aller Kennzahlen, die aktualisiert werden müssen (insbesondere werden solche Kennzahlen identifiziert, die mit den ausgewählten Kennzahlen rechentechnisch verknüpft sind),
- Bestimmung aller Basisdatentabellen, die aktualisiert werden müssen (da die Kennzahlenwerte nichts anderes als Aggregationen der Basisdaten sind, müssen alle entsprechenden BD-Tabellen aktuell sein, bevor man mit der Berechnung der Kennzahlenwerte beginnen kann),
- Bestimmung aller Datenquellen, von denen die Daten angefordert werden müssen (die BD-Tabellen beinhalten Kopien der operationalen Datenquellen, deshalb müssen zu ihrer Aktualisierung die Änderungen der operationalen Daten angefordert werden).

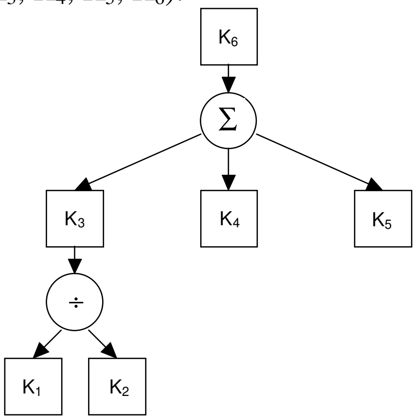
Nachdem alle diese Analysen durchgeführt worden sind, kann das System zu der eigentlichen Migration übergehen. Die Prozesse rechts in der Abbildung 3-8 füllen die Puffer, aktualisieren die Basisdaten und berechnen schließlich die Kennzahlenwerte. Weshalb diese Schrittreihenfolge notwendig ist, haben wir bereits in dem Abschnitt 3.4 "Entwicklung der DW-Datenbank" erklärt. Zur Erinnerung und für die direkte Assoziation haben wir auch das Schichtenmodell der DB-Entitäten in die Abbildung 3-8 integriert (kleiner Kasten links unten).

Nach dieser kurzen Beschreibung des Data-Migration-Systems werden wir im folgenden alle Prozesse in der Folge des Kontrollflusses durchgehen und die notwendigen Aktionen erklären. Bei den Prozessen, die mit den operationalen Daten zu tun haben, werden wir eine Stufe tiefer steigen und ihren Aufbau näher beschreiben.

Am Anfang steht die Analyse des Kennzahlengraphen (Bubble 2 "KZ_Graph_Analyse"), in der alle zur Aktualisierung erforderlichen Kennzahlen bestimmt werden. Das Ergebnis der Analyse ist eine geordnete und erweiterte Liste von Kennzahlen, in der die Kennzahlen nach ihrer späteren Aktualisierungsreihenfolge angeordnet sind. Das bedeutet, daß die Abhängigkeiten in Rechenstrukturen erkannt werden und diejenigen Kennzahlen am Anfang der Liste eingetragen werden, die eine Basis für die Berechnung anderer Kennzahlen sind.

Ein Beispiel dazu:

Sei eine Rechenstruktur gegeben (siehe Abbildung). Der Benutzer interessiert sich für die aktuellen Werte der Kennzahl K_3. Der Prozeß "KZ_Graph_Analyse" (Abbildung 3-8) berechnet eine geordnete Liste (K_1, K_2, K_3, K_4, K_5, K_6).



Die Berechnung der Kennzahlen K_1 und K_2 ist zweifellos ein Muß für die Berechnung der Kennzahl K_3. Etwas schwieriger ist es die Notwendigkeit der Aktualisierung der Kennzahlen K_6 , K_4 und K_5 zu erklären.

Nun, wir fangen mit der Kennzahl K_6 an. Stellen wir uns eine Situation vor, wo infolge der Aktualisierung ein Wert der Kennzahl K_3 geändert wurde. Nach der Aktualisierung will der Anwender die Werte der Kennzahl K_6 einsehen und später auch die Werte von K_3. Ihm werden K_6-Werte präsentiert , die noch immer auf den alten Werten von K_3 aufbauen, obwohl sich diese verändert haben. Die Situation ist irreführend für den Benutzer, da er mit zwei unterschiedlichen Zeitständen der Daten konfrontiert wird.

Somit müssen bei der Kennzahlenaktualisierung auch alle übergeordneten Kennzahlen in den Aktualisierungsvorgang einbezogen werden. Da aber die Spitzenkennzahlen ihrerseits die Aktualität aller untergeordneten Kennzahlen fordern, müssen schließlich alle Kennzahlen der gesamten Rechenstruktur neuberechnet werden. Nur so bleibt die Konsistenz des Systems erhalten.

Das obige Beispiel stellt die Frage auf, was denn der Vorteil der Kennzahlenauswahl sei, wenn am Schluß doch alle Kennzahlen aktualisiert werden müssen. Die Antwort auf die Frage wurde bereits gegeben - es werden die "Nachbarn" in einer Rechenstruktur aktualisiert, nicht jedoch die Nachbarn in anderen Strukturtypen. Kennzahlen, die durch Ordnungs- , heuristische und empirisch-statistischen Strukturen verbunden sind, betrachtet das System bei der Aktualisierung als unabhängig.

Eine Besonderheit hat die Aktualisierung der Kennzahlen, die als Sichten implementiert sind. Dort ist eigentlich nichts zu machen, da die Werte solcher Kennzahlen erst zum Zeitpunkt einer Abfrage berechnet werden und deshalb "immer" aktuell sind. Dementsprechend sind die Aktualisierungsvorschriften dieser Kennzahlen "leer", sie führen keine Operationen auf der Datenbank aus. An der Berechnung der "Geordnete_und_erweiterte_Liste_von_KZ" ändert sich aber nichts, die Kennzahlen K_1, K_2, K_3, K_4, K_5 und K_6 aus dem Beispiel müssen in dem gleichen Umfang und gleicher Reihenfolge berechnet werden, wie früher, denn die Argumentation aus dem Beispiel behält auch hier ihre Gültigkeit.

Der Datenfluß “Geordnete_und_erweiterte_Liste_von_KZ” wird zum Prozeß “Datengraph_Analyse_BD_Tab_Bestimmung[30]” geleitet (Bubble 3 in Abbildung 3-8). Hier erfolgt die Analyse des Datengraphen.

Analog zu der Kennzahlengraph-Analyse werden bei der Datengraph-Analyse die Basisdatentabellen, deren Aktualität eine Voraussetzung für die Neuberechnung von Kennzahlenwerten ist, bestimmt und in eine Liste eingetragen. Diese Liste ist nicht geordnet, da die Basisdatentabellen zyklische Abhängigkeiten aufweisen können und deshalb eine Vorgabe der Aktualisierungsreihenfolge nicht möglich ist. Die Zyklen können entstehen, weil das Datenmodell der Basisdaten sich nicht auf eine einfach Hierarchie reduzieren läßt, wie es der Fall bei den Kennzahlen ist.

Der Datenstrom, der den Bubble “Datengraph_Analyse_BD_Tab_Bestimmung” verläßt, erreicht den Prozeß “Datengraph_Analyse_Dat_Ql_Bestimmung[31]” (Bubble 4 in der Abbildung 3-8). Hier wird der Datengraph nochmals analysiert, und zwar mit dem Ziel, die Datenquellen zu bestimmen, die die Daten für Basisdatentabellen liefern. Das Ergebnis ist eine Liste von Datenquellen, von denen man die Daten anfordern wird.

Mit dem Prozeß “Datengraph_Analyse_Dat_Ql_Bestimmung” endet der Vorbereitungsteil der Datenmigration, ab jetzt beginnt die Phase der Datenextraktion, -umformung und -speicherung.

3.6.3 Datenanforderung

Wir steigen eine Stufe tiefer in der Hierarchie unserer Analyse und untersuchen genauer den Prozeß “Datenanforderung” aus der Abbildung 3-8. Das Ziel dieses Prozesses ist die Anforderung der Daten von den Datenquellen und ihre Abspeicherung in den Puffern der Datenbank.

Die Liste der abzufragenden Datenquellen, die wir im vorherigen Abschnitt als letzte behandelten, ist einer der Input-Parameter des Prozesses “Datenanforderung”. Anhand dieser Liste fordert der Prozeß die Daten von den Datenquellen an. Das DFD “Datenanforderung” (Abbildung 3-9) zeigt diesen Vorgang genauer.

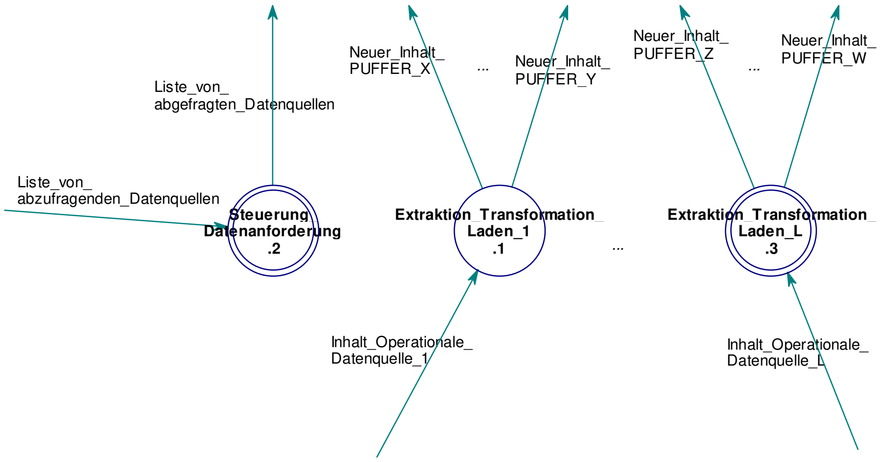


Abbildung 3-9 DFD “Datenanforderung”

---

[30] Eine Abkürzung für “Datengraphanalyse : Basisdatentabellen-Bestimmung”

[31] Eine Abkürzung für “Datengraphanalyse : Datenquellen-Bestimmung”

Für jede in der Liste eingetragene Datenquelle wird ein eigener Migrationsprozeß gestartet. In der Abbildung 3-9 sieht man den Prozeß "Steuerung_Datenanforderung", der die insgesamt L Migrationsprozesse sequentiell oder auch parallel startet. In dieser Arbeit wollen wir keine festen Vorgaben bezüglich des Aufbaus dieser Prozesse geben - wegen der Vielfältigkeit der Datenquellen ist es gar nicht möglich, auf alle Einzelheiten einzugehen. Nur als Beispiel zeigen wir einen möglichen Aufbau eines Migrationssprozesses (Abbildung 3-10). Wir stellen an diese Prozesse nur eine einzige Forderung : sie müssen die Daten aus den operationalen Quellen in die Warehouse-Datenbank kopieren. Dafür stellen wir eine geeignete Menge von Puffertabellen zur Verfügung.

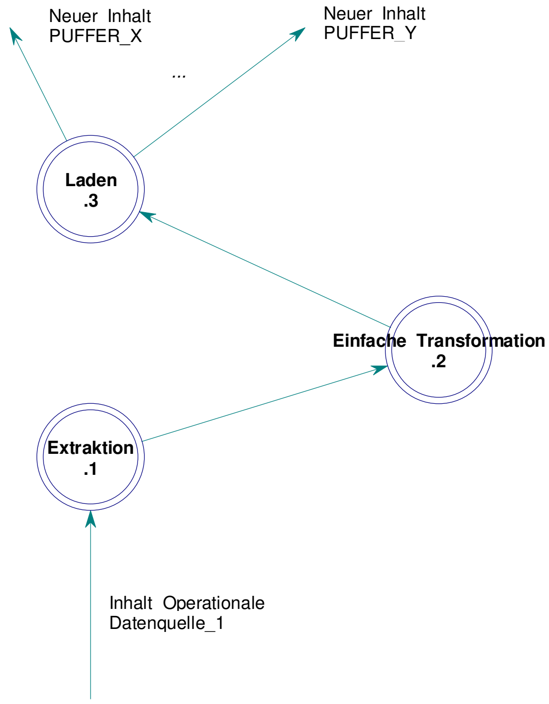


Abbildung 3-10 DFD "Extraktion_Transformation_Laden_1"

Die Realisierung der Datenextraktion ist abhängig von dem Typ der Datenquelle. Für eine relationale Datenbank kann sie z.B. mit einem separaten Programm durchgeführt werden, das den Inhalt einer oder mehreren Tabellen der Datenbank abfragt und ihn (den Inhalt) in temporären Dateien des Dateisystems oder direkt in dem Data Warehouse abspeichert. Alternativ kann man auf die Extraktions-Werkzeuge der Datenbank zurückgreifen.

Sehr wichtig ist die Frage, welche Daten aus den Datenquellen übernommen werden sollen. Ideal wäre der Fall, wo nur die Änderungen kopiert werden könnten - dadurch würde man Zeit und Ressourcen sparen. In manchen Fällen ist das überhaupt nicht oder nur schwer möglich (z.B. bei den flat files). Dann muß das System den kompletten Inhalt übernehmen und erst in dem Warehouse eine sog. Delta-Berechnung[32] durchführen.

Die Transformation, die in dieser Phase erfolgt, sollte höchstens im Formatieren der rohen Daten bestehen. Wir schlagen vor, die eigentliche Transformation erst dann durchzuführen, wenn die Daten in die Warehouse-Datenbank geladen sind. Es ist dann möglich, die notwendigen Joins der Tabellen und andere Umformungen mit Hilfe der vorhandenen Mitteln der Da-

[32] Delta-Berechnung - Bestimmung der Änderungen in den Daten über den direkten Vergleich von zwei unterschiedlichen Datenständen.

tenbank zu realisieren. Manuelle Programmierung dieser Routinen wäre vergleichbar mit der Neuerfindung des Rades.

Das Laden der Daten in die Datenbank kann wiederum durch separate Programme oder Lade-Tools des Warehouse-DBMS erfolgen. Es werden Puffertabellen gefüllt (im Allgemeinen mehrere pro Datenquelle), die eine Struktur aufweisen, die sehr ähnlich der Struktur von Datenquellen ist. Beispiel:

Der Trap-Monitor des Werkzeugs HP OpenView erzeugt eine Logdatei (flat file) mit einem Eintrag pro angekommene Trap. Diese Datei wird durch Anhängen von neuen Einträgen an das Dateiende ständig aktualisiert:

```
…
835252607 1 Thu Jun 20 08:36:47 1996 xwslug.lrz-muenchen.de   M Node up;1 17.1 6 58916864 62675
835252694 1 Thu Jun 20 08:38:14 1996 sunbroy22.informatik.tu-muenchen.de N IF 131.159.0.13 up;1 17.1 6 58916866 29524
…
```

Die Daten im Originalformat sind zum Laden in die Datenbank nicht geeignet, sie können nur als Ganzes, d.h. ohne eine Teilung in Felder, geladen werden. Es wäre angebracht, die Daten durch geeignete Transformationen der Zeilen in ein Standard-DB-Ladeformat zu bringen (z.B. “comma separated” oder “quoted strings”, wie im Beispiel unten). Dann könnten die Standard-Lade-Werkzeuge des Warehouse-DBMS verwendet werden, die deutlich schneller sind, als ESQL-Programme:

```
…
835252607 1 “Thu Jun 20:08:36:47 1996” “xwslug.lrz-muenchen.de” “M Node up” 1 “17.1” 6 58916864 62675
835252694 1 “Thu Jun 20:08:38:14 1996” “sunbroy22.informatik.tu-muenchen.de” “N IF 131.159.0.13 up” 1 “17.1” 6 58916866 29524
…
```

Die Puffertabelle, die diese leicht transformierten Daten aufnehmen soll, könnte folgendermaßen definiert werden:

```
CREATE TABLE PUFFER_TRAPSTAT (
    TIMESTAMP       NUMBER,
    ID              NUMBER,
    TRAP_DATUM      CHAR(24)
    AGENT           VARCHAR(255),
    TRAP_INFO       VARCHAR(255),
    …)
```

Man kann im Allgemeinen nicht davon ausgehen, daß alle operationalen Quellen zum Zeitpunkt der Aktualisierung verfügbar sind. So können z.B. die DB-Server-Prozesse runtergefahren sein (zu Wartungszwecken oder wegen Fehler), das Netz-Dateisystem kann den Zugriff auf Log-Dateien verweigern (wiederum wegen Wartung oder Fehler) usw. Es ist deshalb notwendig, eine Liste der tatsächlich abgefragten Datenquellen an nachfolgende Prozesse weiterzureichen (Data Flow “Liste_von_abgefragten_Datenquellen” in Abbildung 3-9 und Abbildung 3-8).

3.6.4 Aktualisierung der Basisdatentabellen

Nach dem Laden der rohen Daten aus den operationalen Datenquellen in die Puffertabellen ist es jetzt möglich die Basisdatentabellen zu aktualisieren (Prozeß “Aktualisierung_ Basisdaten_Tabellen” in Abbildung 3-8). Wir zeigen den genaueren Ablauf dieses Prozesses in der Abbildung 3-11.

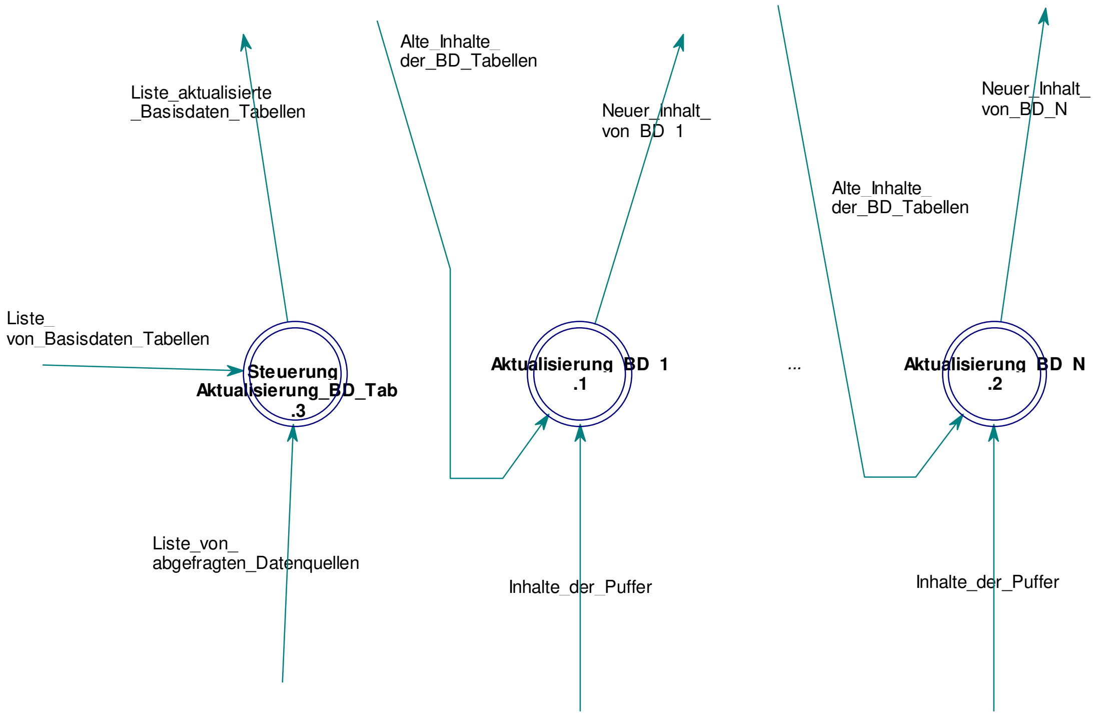


Abbildung 3-11 DFD "Aktualisierung_BD_Tabellen"

Die Aktualisierung der Basisdatentabellen läuft größtenteils einheitlich ab. Man kann deshalb den gesamten Vorgang in mehrere Subprozesse einteilen, die die Aufgabe der Aktualisierung einer einzigen Tabelle übernehmen. Zusätzlich wird eine Steuerungsfunktionalität gebraucht, welche die Aktualisierungsprozesse startet.

Es existiert deshalb ein Steuerungsprozeß ("Steuerung_Aktualisierung_BD_Tab"), der als Input-Parameter zwei Listen erhält : "Liste_von_Basisdaten_Tabellen" und "Liste_von_ abgefragten_Datenquellen". Der erste Parameter stammt von der Datengraph-Analyse und ist eine "Wunschliste" - er enthält die Namen von Basisdatentabellen, die für die Kennzahlenaktualisierung notwendig sind und deshalb auch selbst aktualisiert werden *sollten*. Der zweite Parameter enthält die Namen der Datenquellen, die zum Zeitpunkt der Datenanforderung verfügbar waren und deshalb abgefragt werden *konnten*. Da u.U. nicht alle Datenquellen abgefragt werden konnten, die für die Berechnung der BD-Tabellen aus der Wunschliste notwendig sind, wird der Steuerungsprozeß manche Aktualisierungen weglassen müssen - nicht alle Aktualisierungswünsche können erfüllt werden. Diejenigen BD-Tabellen, die aktualisiert werden konnten, werden als ein Output-Parameter ("Liste_aktualisierte_Basisdaten_Tabellen") an weitere Prozesse weitergeleitet. Ohne diese Liste kann der Aktualisierungsprozeß der Kennzahlen nicht entscheiden, welche Kennzahlen aktualisiert werden können (dort herrscht die gleiche Situation wie hier).

Die Basisdatentabellen können nicht einfach dadurch aktualisiert werden, daß man ihre Inhalte vollständig löscht und anschließend neuberechnet. Die operationalen Datenquellen haben unter Umständen eine sehr kurze Aufbewahrungsfrist[33] für Daten, so daß die Kennzahlen nur über einem kurzen Zeitraum berechnet werden könnten. Aus diesem Grund müssen wir die Aktualisierung der Basisdaten inkrementell gestalten, d.h. über die Operationen "Einfügen" und "Ändern", nicht aber mit "Löschen".

[33] Das gilt besonders für die Log-Dateien - sie werden monatlich oder sogar wöchentlich gelöscht.

Dazu müssen die “alten” Inhalte der Basistabellen (Data Flow “Alte_Inhalte_BD_Tabellen”) mit den aktuellen Inhalten der Puffer (“Inhalt_der_Puffer”) verglichen und daraus die neuen “Delta”-Sätze gebildet werden. Eine mögliche Aktualisierungsprozedur könnte z.B. so aussehen:

```
PROCEDURE Aktualisiere_BD_1 IS
BEGIN

   /* Überprüfe alle Datensätze im Puffer */
   FOR x IN puffer_1 LOOP

      /* Suche ein y in der Basisdatentabelle, das dem x "entspricht"*/
      /* Was die Relation <entspricht> im einzelnen bedeutet, hängt */
      /* von den konkreten Basisdaten-Tabelle und Puffer-Tabelle ab.*/
      FIND y IN bd_1 WHERE y <entspricht> x;

      IF y NOT AVAILABLE THEN

         /* Der Datensatz ist neu, deshalb erzeuge */
         /* einen neuen Eintrag in der Basisdatentabelle */
         INSERT x INTO BD_1;

      ELSE

         /* Der Datensatz x ist vorhanden, deshalb ändere */
         /* den alten Eintrag in der Basisdatentabelle */
         UPDATE y SET x;

      END IF;

      /* Lösche x im Puffer */
      DELETE x;

   END LOOP;

END;
```

3.6.5 Aktualisierung der Kennzahlen-Wertetabellen

Nach der Datenanforderung und der Aktualisierung der Basisdatentabellen sind wir jetzt zu der Aktualisierung der Kennzahlen gekommen. Die Abbildung 3-12 ist das Data Flow Diagramm dieses Prozesses.

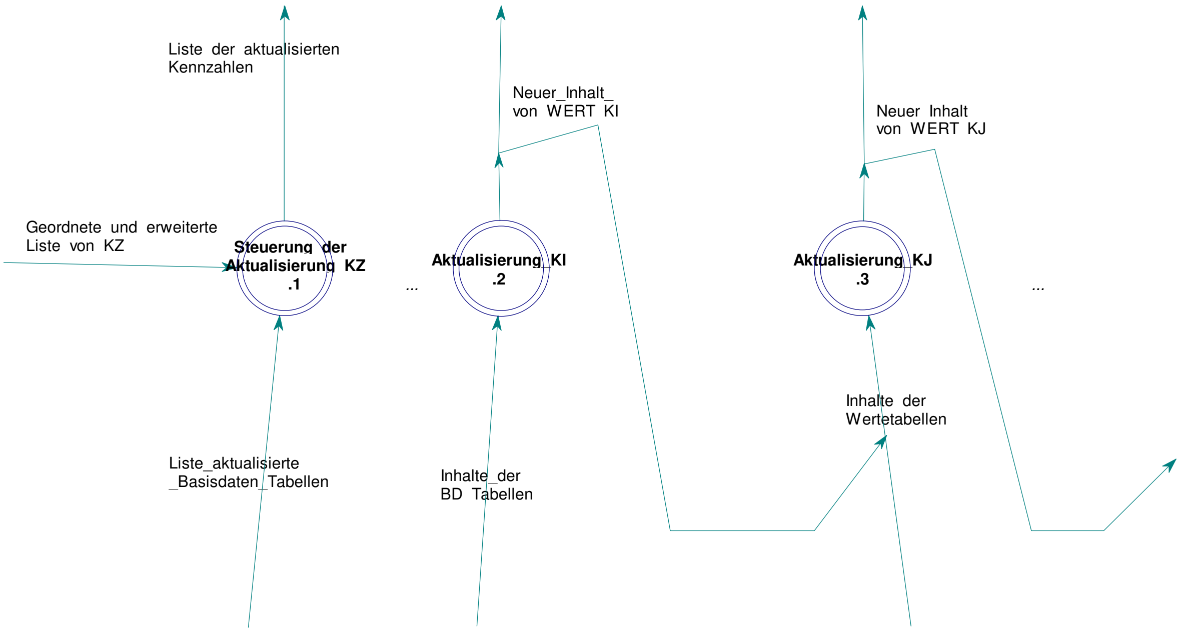


Abbildung 3-12 DFD “Aktualisierung_Kennzahlen”

Wie auch bei der Aktualisierung der Basistabellen, kann dieser Prozeß in mehrere Subprozesse zerlegt werden, in denen nur eine Kennzahl aktualisiert wird. Auch hier gibt es einen Steuerungsprozeß, der die einzelnen Kennzahlenaktualisierungsprozesse startet. Die Menge der Kennzahlen, die aktualisiert werden sollte (Wunschliste "Geordnete_und_erweiterte_Liste_von_KZ") wird um solche Kennzahlen bereinigt, für welche die Voraussetzungen für die Aktualisierung fehlen (d.h. ordnungsgemäß aktualisierte Basisdatentabellen).

Der Prozeß der Kennzahlenaktualisierung unterscheidet sich von der Basisdaten-Aktualisierung insofern, daß die Kennzahlen nach einer bestimmten Reihenfolge berechnet werden. Ein weiterer Unterschied besteht darin, daß wir hier die Möglichkeit haben, die Kennzahlenaktualisierung einfach durch Löschen und Neuberechnung zu realisieren. Die Basisdatentabellen enthalten eine vollständige Historie der operationalen Daten und erlauben deshalb, die gelöschten Kennzahlenwerte bei der Neuberechnung wiederherzustellen.

Zuerst werden terminale Kennzahlen aktualisiert, d.h. solche Kennzahlen, die direkt aus den Inhalten der Basisdatentabellen berechnet werden können (in Abbildung 3-12 ist so eine Kennzahl durch den Bubble "Aktualisierung_KI" repräsentiert). Auf der Basis ihrer Werte werden anschließend die nichtterminalen Kennzahlen (in der Abbildung 3-12 durch den Bubble "Aktualisierung_KJ" dargestellt) aktualisiert.

Als Beispiel zeigen wir eine Aktualisierungsvorschrift der nichtterminalen Kennzahl K2 "Sachgebiete monatlich" (definiert im Anhang A):

```
PROCEDURE Aktualisiere_K2 IS
  CURSOR curNeueWerte IS
    SELECT sachgebiet1 s1,
      sachgebiet2 s2,
      sachgebiet3 s3,
      TRUNC(stunde,'MM') monat,
      SUM(wert) wert
    FROM wert_k1
    GROUP BY sachgebiet1,sachgebiet2,sachgebiet3,TRUNC(stunde,'MM');

BEGIN

  DELETE FROM WERT_K2;

  FOR recNeuerWert IN curNeueWerte LOOP

    INSERT INTO WERT_K2
      (wert_id, sachgebiet1, sachgebiet2,sachgebiet3,monat, wert)
    VALUES(k2_wert_id_seq.NEXTVAL, recNeuerWert.s1, recNeuerWert.s2,
      recNeuerWert.s3, recNeuerWert.monat,recNeuerWert.wert);

  END LOOP;

  COMMIT;

END;
```

Im Beispiel werden zuerst die alten Werte von K2 gelöscht. Danach werden die neuen Werte der Kennzahl K2 berechnet und nacheinander in die Wertetabelle eingestellt. Anschließend werden die Änderungen bestätigt. Der Aufbau der Aktualisierungsprozeduren ist bei allen Kennzahlen derselbe, der einzige variierende Teil ist die Definition der SQL-Berechnungsvorschrift (sog. Cursor).

Wie bereits im Abschnitt 3.6.2 erwähnt, sind die Aktualisierungsvorschriften der Kennzahlen, die als Sichten realisiert sind, leer. Die Werte dieser Kennzahlen werden erst zum Zeitpunkt einer Abfrage berechnet, man braucht deshalb während der Warehouse-Aktualisierung keine DB-Operationen durchzuführen.

Mit der Aktualisierung der Kennzahlen ist der Prozeß der DW-Aktualisierung abgeschlossen. Der Benutzer kann die Kennzahlenwerte z.B. über ein kommerzielles Abfrage-Tool von dem DW-System anfordern und für seine Arbeit verwenden.

3.7 Zusammenfassung

Das Kapitel 3 beschrieb die Realisierungsmethode des Frameworks zum Aufbau von technischen RZ-Kennzahlensystemen. Wir schlugen vor, das Kennzahlensystem auf der Grundlage eines Data Warehouses zu bauen. Gemäß dem Referenzmodell eines Data Warehouses gliederten wir die Vorstellung der Methode in drei Teile - Entwicklung der DW-Datenbank, Entwicklung des Metadaten-Repository und Entwicklung des Data-Migration-Systems.

Für die Datenstrukturen der Datenbank gaben wir ein Schichtenmodell an, das aus drei Typen von Entitäten besteht : den Puffern, den Basisdaten- und den Wertetabellen. Im Metadaten-Abschnitt identifizierten wir die wichtigsten Klassen von Metadaten und gingen auf die wichtigsten von ihnen ein. Insbesondere wurde der sog. Datengraph eingeführt, der u.a. die Informationen über die Datenquellen beinhaltet. Im Abschnitt "Entwicklung des Data-Migration-Systems" wurde die Strukturierte Analyse des Migrations-Problems durchgeführt. Die wichstigsten Ergebnisse dieser Analyse sind die inkrementelle Aktualisierung der Basisdaten und die Auswahl-Möglichkeit der zu aktualisierenden Kennzahlen.

4. Ein Prototyp des technischen Kennzahlensystems für das Leibniz-Rechenzentrum

Die Kapiteln 2 und 3 bilden eine Grundlage für die Entwicklung eines technischen Kennzahlensystems für das Leibniz-Rechenzentrum. In dem vierten Kapitel dieser Arbeit beschreiben wir den Prototyp des Kennzahlensystems, der für das LRZ gebaut wurde.

Zuerst gehen wir auf die Rahmenbedingungen des LRZ-Kennzahlensystems ein. Die Definition von konkreten Kennzahlen und Kennzahlenstrukturen erfolgt im zweiten Abschnitt. In dem dritten Unterkapitel beschreiben wir die Implementierung des Kennzahlen-Data-Warehouses, in dem die LRZ-Kennzahlen berechnet werden.

4.1 Die Rahmenbedingungen des Projektes

Der Rahmen für dieses Projekt bestand einerseits aus den zur Verfügung stehenden Werkzeugen zum Bau des Data Warehouses und andererseits aus den vorgegebenen operationalen Datenquellen. Außerdem spielten auch die sehr knappen personellen Ressoucen, die für die Implementierung zur Verfügung standen (ca. 1,5 Mann-Monate), eine restriktive Rolle.

Ein kommerzielles Projekt, in dem ein Data Warehouse gebaut wird, dauert in der Regel nur wenige Monate, jedoch werden in ihm Dutzende von Mitarbeitern involviert. Es werden spezielle technische Mittel eingesetzt, wie z.B. dedizierte Metadaten-Repositories, Data-Migration-Systeme und DB-Systeme (z.B. von den Herstellern Prism, Platinum, Carleton, ETI, Information Builders u.a.).

Unser Prototyp, der ohne diese Tools auskommen mußte, kann deshalb nur als eine Studie der Implementierbarkeit der Konzepte betrachtet werden, die in den Kapiteln 2 und 3 vorgestellt wurden. Bei der Weiterentwicklung des Prototyps sollten einige dieser Werkzeuge unbedingt verwendet werden. Dadurch wird es möglich sein, mehr Datenquellen in das Data Warehousing einzubeziehen, als wir es tun konnten.

4.1.1 Verfügbare Implementierungswerkzeuge

Zur Implementierung standen uns folgende Werkzeuge zur Verfügung:

1) Relationales DB-Managementsystem *Oracle Version 7*,
2) Entwicklungsumgebung *Developer/2000* von Oracle,
3) C++-Compiler *sunpro* von Sun,
4) ESQL-Präprozessor *proc* von Oracle in der Version 2.1,
5) Modellierungswerkzeug *Promod* der Firma debis,
6) Testversion des Modellierungswerkzeuges *With Class* der Firma MicroGold Software,
7) Klassenbibliothek *LEDA* des Max-Planck-Instituts für Informatik, Saarbrücken,
8) Unterschiedliche API's und Bibliotheken von Oracle, Hewlett Packard und Remedy.

Das Datenbanksystem Oracle7 dient in dem Prototyp als Datenspeicher sowohl für die Kennzahlenwerte und Basisdaten, als auch für die Haltung der Metadaten. Mit der Entwicklungsumgebung Developer/2000 sind die Stored Procedures entwickelt, welche die Aktualisierung der Werte- und Basisdatentabellen durchführen. Das Data-Migration-System wurde in der Sprache C++ mit Embedded SQL programmiert, mit dem Präprozessor proc verarbeitet und mit dem Compiler sunpro übersetzt. Bei der Programmierung verwendeten wir die Klassenbibliothek LEDA, in der die von uns benötigten Klassen wie "Graph", "Liste" und viele andere bereits implementiert sind. Für den Zugriff auf die operationalen Daten benutzten wir die API's und Bibliotheken der jeweiligen Hersteller.

Die Modellierung der relationalen Daten (ER-Modellierung) erfolgte mit Promod. Dasselbe Werkzeug setzten wir auch bei der Strukturierten Analyse ein. Bei dem objektorientierten Design des Migrationssystems verwendeten wir "With Class".

4.1.2 Vorgegebene Datenquellen

Für die prototypische Implementierung des LRZ-Kennzahlensystems wurden uns folgende Datenquellen vorgegeben :

1) Action Request System (das Trouble-Ticket-System) von der Firma Remedy,
2) Log-Dateien von snmpCollect (Statistiken der MIB-Variablen),
3) Log-Datei des Trap-Dämons,
4) Log-Dateien der Ascend Access-Server,
5) Datenbank von HP OpenView,
6) Netz-Dokumentation.

Das AR-System wird zur Erfassung von Trouble-Tickets und zur Kontrolle von deren Bearbeitung eingesetzt. Es ist eine vergleichsweise kleine Datenbank mit ca. 8000 Einträgen (Stand März 1997).

Das Tool snmpCollect dient der Protokollierung von Werten der MIB-Variablen. Derzeit werden ca. 50 MIB-Agenten angepollt und ein Dutzend Variablen[34] abgefragt. Die gewonnenen Daten werden in Log-Dateien abgelegt (eine pro Variable).

Der Trap-Dämon (trapd) sammelt die im LRZ-Netz anfallenden Trap-Meldungen. Zur Zeit sind es ca. 10.000 Traps pro Tag. Dieser Trap-Dämon führt eine Log-Datei, diese wird aber wg. Platzgründen immer wieder gelöscht.

Die am LRZ eingesetzten Access-Server der Firma Ascend protokollieren die eingehenden Anrufe auch in Log-Dateien. Es existieren zusammenfassende und detaillierte Logs. Wegen der anfallenden Datenmengen (ca. 30 MB pro Woche) sind nur die Logs der letzten sechs Wochen verfügbar.

Die OV-Datenbank beinhaltet eine Fülle von Informationen über das LRZ-Netz (z.B. die Netz-Topologie, Traps usw.). Ergänzend dazu existiert eine separate Netzdoku-Datenbank, in der z.B. die Kabelpläne hinterlegt sind.

[34] Variablenreihen, wie ifInOctets.1 bis ifInOctets.602, betrachten wir zur Vereinfachung als eine einzige Variable.

Soweit es möglich war, haben wir für die erwähnten Datenquellen ein Formular ausgefüllt, in dem die für Data Migration wichtigen Informationen enthalten sind. Diese Formulare können im Anhang C eingesehen werden.

4.2 Kennzahlensystem i.e.S.

Als erster Schritt wurden in dieser Arbeit zwei Datenquellen in das Data Warehousing am LRZ einbezogen: das AR-System und die OV-Datenbank. Nicht alle Daten wurden aus diesen Datenbanken übernommen : in ARS waren es nur die Trouble-Tickets, in OV - nur die Ausfälle der Geräte.

Die Daten aus dem AR-System erlauben es, für die Dienstgruppe "Beratung & Unterstützung" die Charakteristika "Nutzung/Auslastung" und "Leistungsgrad" mit Kennzahlen zu beschreiben. Außerdem existiert die Möglichkeit, das Charakteristikum "Supportintensität" der technischen Betriebsmittel zu überwachen (Abbildung 4-1).

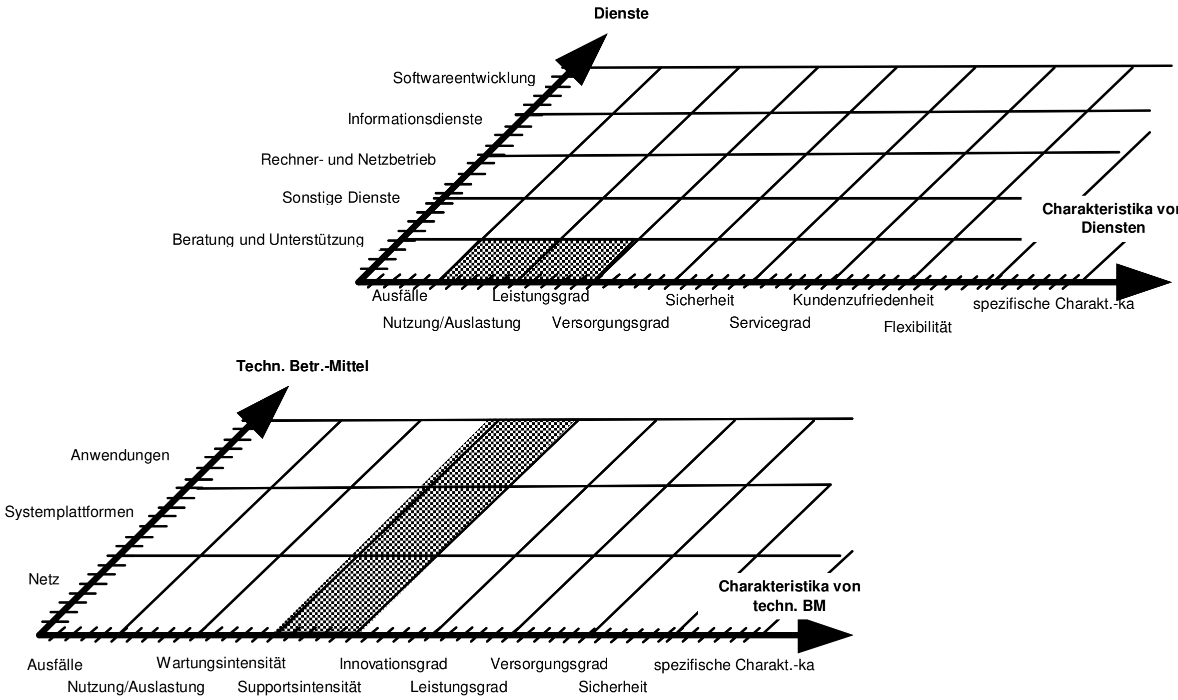


Abbildung 4-1 Aus ARS-Daten definierbare Kennzahlen

Die aus OV übernommenen Ausfalldatensätze sind eine Grundlage für die Kennzahlengruppe "Ausfälle der Technischen Betriebsmittel" (Abbildung 4-2):

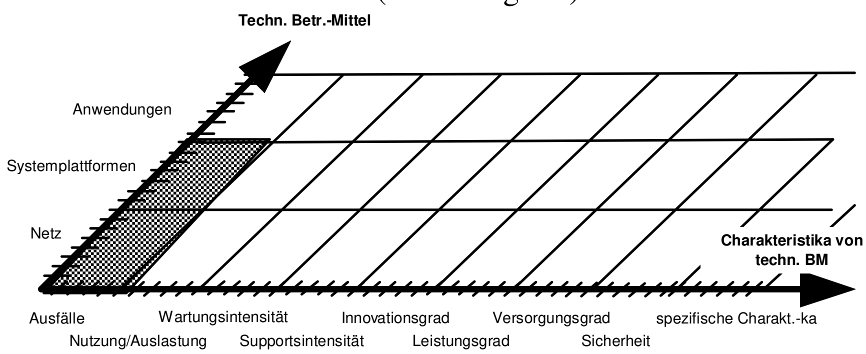


Abbildung 4-2 Aus OV-Daten definierbare Kennzahlen

4.2.1 Kennzahlen

Folgende Kennzahlen erschienen uns als sinnvoll:

- K1 - Sachgebiete stündlich,
- K2 - Sachgebiete monatlich,
- K3 - ∅ Bearbeitungsdauer,
- K4 - Spitzenbedarfszeiten,
- K5 - TT-Eingang täglich,
- K7 - TT-Eingang monatlich,
- K8 - Verfügbarkeit 8-18 täglich,
- K9 - Verfügbarkeit 0-24 täglich,
- K10 - LRZ-Verfügbarkeit 8-18 monatlich,
- K11 - LRZ-Verfügbarkeit 0-24 monatlich,
- K12 - Modell-Verfügbarkeit 0-24 monatlich,
- K13 - Verantwortlicher-Verfügbarkeit 8-18 monatlich

(die Nummer K6 wurde nicht vergeben).

Die Kennzahlen K1 und K2 liefern Informationen darüber, wieviele Trouble-Tickets eines bestimmten Sachgebiets innerhalb einer Stunde bzw. eines Monats erfaßt wurden. Da die Sachgebiete eine maximal dreistufige Hierarchie bilden, haben wir bei jeder Kennzahl, die mit Sachgebieten zutun hat, die drei Parameter "Sachgebiet1", "Sachgebiet2" und "Sachgebiet3" eingeführt. Die Aufgabe dieser zwei Kennzahlen ist es, die Untersuchung der Trouble-Tickets-Verteilung bzgl. der Sachgebiete zu ermöglichen. Insbesondere können mit Hilfe dieser Kennzahlen solche Sachgebiete identifiziert werden, die den größten oder niedrigsten Anteil an der Gesamtzahl der Trouble-Tickets haben. In April 1997 waren es die "Netzdienste"-Trouble-Tickets, die mit über 50 % (147 von 289) die Spitzenposition hielten.

Die Kennzahl K3 "∅ Bearbeitungsdauer" beantwortet die Frage: "Wie lange werden Trouble-Tickets eines Sachgebiets durchschnittlich bearbeitet?". Der Durchschnitt wird über alle Trouble-Tickets eines Monats gebildet, deshalb besitzt diese Kennzahl zusätzlich zu den drei Sachgebiete-Parametern noch den Parameter "Monat". Wir haben diese Kennzahl sachgebietsbezogen gemacht, da es zu erwarten ist, daß die Probleme aus verschiedenen Sachgebieten unterschiedlich lange bearbeitet werden. Diese Kennzahl ist ein Indikator für die Qualität der Beratung. So konnten wir z.B. feststellen, daß die durchschnittliche Bearbeitungszeit der PC-Benutzerinterface-Probleme seit November 1996 und bis heute ca. 5,5 Kalendertage beträgt (die Werte schwanken von 4.7 bis 5.9 Tagen). Das interessanteste in der Analyse ist es, daß vor November 1996 die Bearbeitungsdauer in der Region von 15-17 Kalendertagen lag.

Auf die Frage "Wieviele Trouble-Tickets werden in einer bestimmten Stunde des Arbeitstages durchschnittlich erfaßt?", gibt die Kennzahl K4 eine Antwort. Auch hier wird der Durchschnitt für einen Monat gebildet. Diese Kennzahl kann zur Bestimmung der Spitzenbedarfszeiten der Hotline verwendet werden. In dem Monat März 1997 waren es die Stunden 11 und 14 Uhr, in denen durchschnittlich die maximale Anzahl der Trouble-Tickets erfaßt wurde.

Die Werte der Kennzahlen K5 und K7 sagen aus, wieviele Trouble-Tickets an einem Tag bzw. in einem Monat erfaßt wurden. Dabei wird keine Unterscheidung in Sachgebiete gemacht. Die Kennzahlen K5 und K7 dienen zur Überwachung der Belastung und Nutzung des Beratungsdienstes. Wir konnten feststellen, daß die Anzahl der monatlich erfaßten Trouble-Tickets zwischen 165 und 300 liegt, und daß die Wochentage Dienstag und Mittwoch die beliebtesten "Beschwerdetage" sind.

Sechs Kennzahlen - K8 bis K13 - beschreiben die Verfügbarkeit der Netze und der Systemplattformen. Dabei wird zwischen der “8-18”- und “0-24”- Verfügbarkeit unterschieden.

Die Kennzahlen K8 und K9 liefern detaillierte Werte über die einzelnen Hosts und Netzkomponenten mit einer Tages-Granularität. Sie können dazu verwendet werden, um Verfügbarkeitsstatistiken für die Plattformen und Netzkomponenten zu erstellen. Geräte, die zu hohe Schwankungen oder zu niedrige Werte aufweisen, sollten vorgemerkt und weiter beobachtet werden.

Die K10 und K11 aggregieren die Verfügbarkeitswerte über einen Monat und für das gesamte LRZ. Der Gedanke bei der Definition dieser Kennzahlen war der, daß man einen akkumulierten Wert für das gesamte Rechenzentrum hat, welcher für den zwischenbetrieblichen Vergleich oder als Nachweis gegenüber den Kunden benutzt werden kann.

Die Kennzahl K12 liefert Statistiken für die Plattformmodelle. Sie könnte z.B. von den Verantwortlichen im Einkauf als die Grundlage für die Kaufentscheidungen dienen.

K13 erlaubt es, die Verfügbarkeit der Plattformen und der Netze nach den verantwortlichen Personen zu unterscheiden. Da es sich in diesem Fall um eine personenbezogene Kennzahl handelt, müssen die gewonnenen Informationen mit äußersten Vorsicht behandelt werden.

Die vollständigen Definitionen der Kennzahlen und ein Beispiel für eine Kennzahlenanalyse findet man in den Anhängen A und B.

4.2.2 Rechenstrukturen und der Datengraph

Die Grundlage für die Berechenbarkeit der im vorigen Abschnitt definierten Kennzahlen bilden die Rechenstrukturen und der Datengraph. Sie sind die wichtigsten Parameter des Data-Migration-Systems, ohne sie ist die partielle Aktualisierung des Data Warehouses unmöglich.

In der Abbildung 4-3 wird die Einordnung der definierten Kennzahlen in die Rechenstrukturen gezeigt. Der zugehörige Datengraph ist dort auch abgebildet. Die Kennzahlen sind hauptsächlich durch Aggregation der Basisdaten und der Werte von anderen Kennzahlen entstanden.

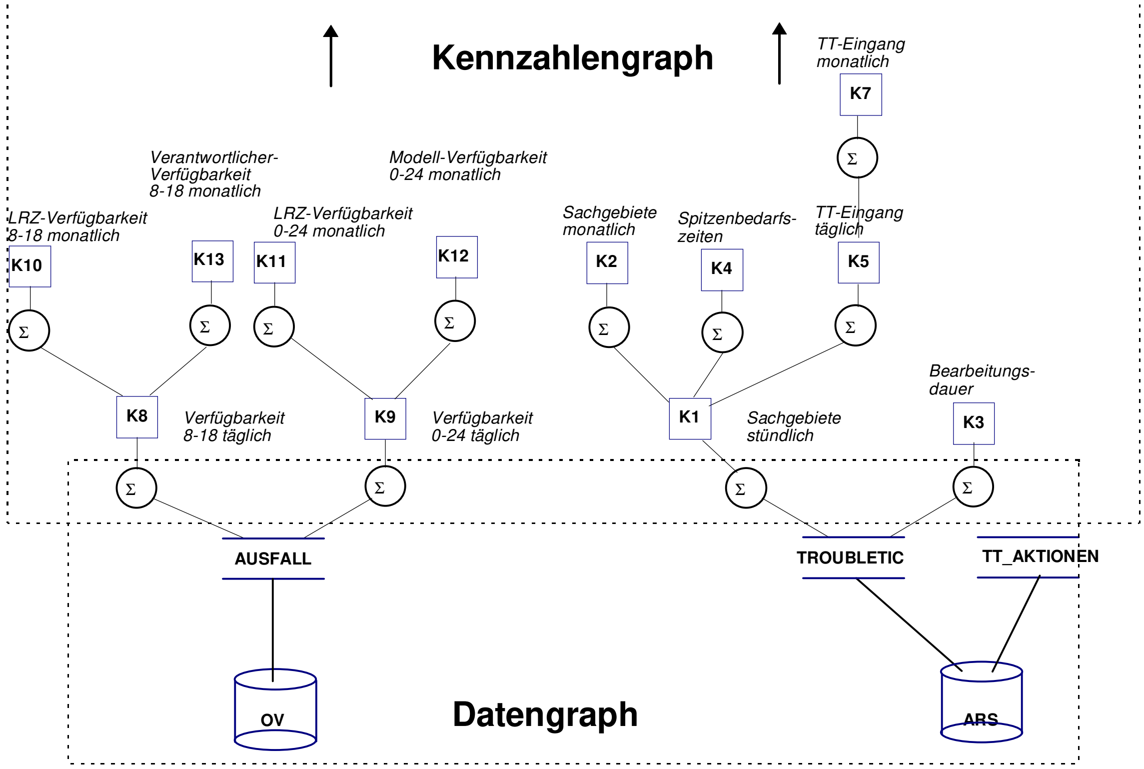


Abbildung 4-3 Rechenstrukturen und Datengraph des LRZ-Kennzahlensystems

Da wir im Prototyp nur zwei Datenquellen in dem Data Warehousing berücksichtigt haben, sind in dem Datengraphen auch nur zwei Datenquellen-Knoten vorhanden ("OV" und "ARS"). Diese Quellen liefern Daten für die Basisdatentabellen AUSFALL, TROUBLETIC und TT_AKTIONEN, von denen momentan nur zwei Entitäten zur Berechnung von Kennzahlen genutzt werden. Die zwei Rechenstrukturen, die auf den Basisdatentabellen AUSFALL und TROUBLETIC aufbauen, sind weitgehend disjunkt - sie haben weder gemeinsame Kennzahlenknoten, noch gleiche Rechenknoten und Kanten. In der Abbildung 4-3 sind nur die untersten Ebenen des Kennzahlengraphen eingezeichnet, oberhalb der Rechenstrukturen werden die Kennzahlen in unterschiedliche Ordnungsstrukturen eingebunden.

4.2.3 Substruktur "Benutzergruppen"

Der Bedarf nach den Kennzahlen hängt u.a. von der Funktion des Benutzers des Kennzahlensystems ab, deshalb definierten wir eine Substruktur, welche die Kennzahlen in drei Gruppen einteilt: die Anwender-, RZ-Mitarbeiter und RZ-Management-Kennzahlen (Abbildung 4-4). Bei Bedarf könnte man diese Gruppen weiter verfeinern.

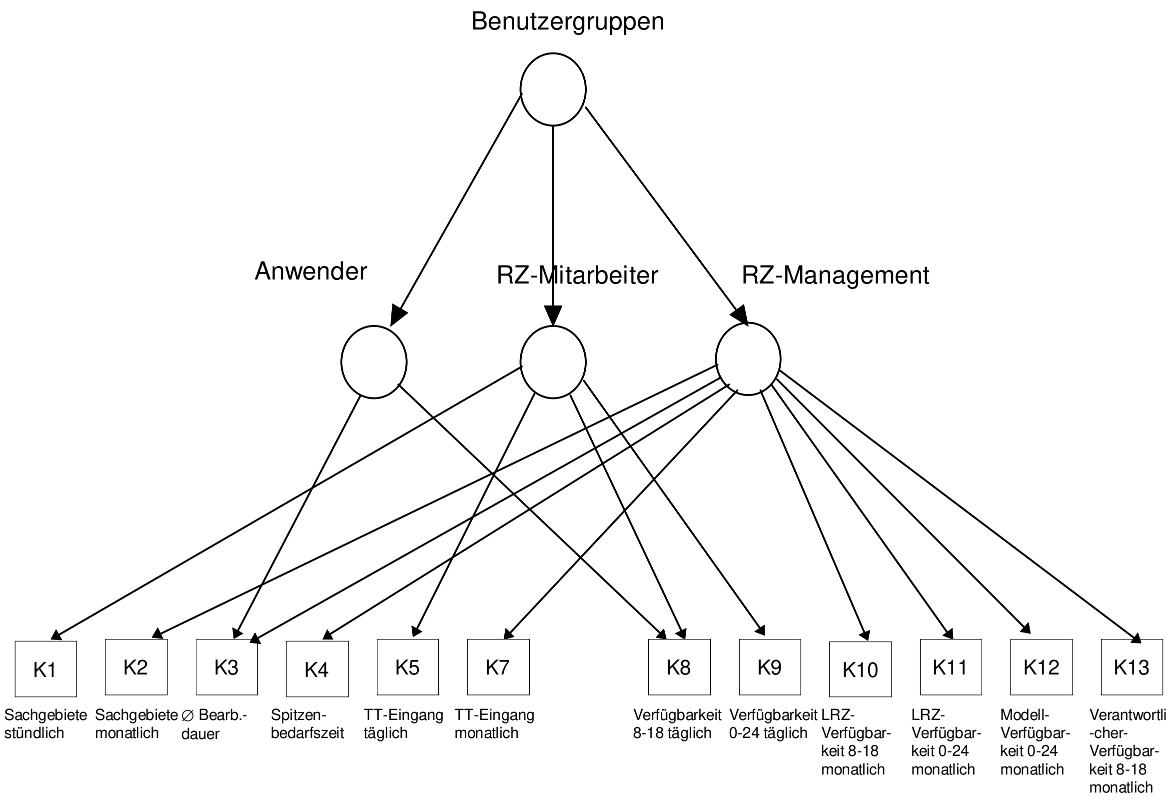


Abbildung 4-4 Substruktur "Benutzergruppen" des LRZ-Kennzahlensystems

Für den Anwender sollten die Kennzahlen K3 "∅ Bearbeitungsdauer" und K8 "Verfügbarkeit 8-18 täglich" (beschränkt auf die Geräte des Anwenders) besonders interessant sein. Sie beantworten ihm die Fragen: "Ist es sinnvoll die Beratungsdienste des LRZ in Anspruch zu nehmen?" bzw. "Wie gut betreut LRZ meine Systeme?".

Die RZ-Mitarbeiter, die für die operative Steuerung verantwortlich sind, werden durch die Kennzahlen K1 "Sachgebiete stündlich", K5 "TT-Eingang täglich", K8 "Verfügbarkeit 8-18 täglich" und K9 "Verfügbarkeit 0-24 täglich" unterstützt. Diese Kennzahlen zeichnen sich durch einen hohen Detaillierungsgrad aus, deshalb stehen den Mitarbeitern sehr ausführliche Statistiken über die eingehenden Trouble-Tickets und über die Verfügbarkeit der einzelnen Plattformen und Netzsegmente zur Verfügung. Bei Bedarf können die Mitarbeiter in den Basisdaten ein noch detaillierteres Wissen vorfinden (alle Ausfälle und Trouble-Tickets werden in dem Data Warehouse dupliziert).

Die hochverdichteten Kennzahlen sind für das RZ-Management bestimmt:

- K2 "Sachgebiete monatlich",
- K3 "∅ Bearbeitungsdauer",
- K4 "Spitzenbedarfszeiten",
- K7 "TT-Eingang monatlich",
- K10 "LRZ-Verfügbarkeit 8-18 monatlich",
- K11 "LRZ-Verfügbarkeit 0-24 monatlich",
- K12 "Modell-Verfügbarkeit 0-24 monatlich",
- K13 "Verantwortlicher-Verfügbarkeit 8-18 monatlich".

Die Kennzahl K2 erlaubt dem RZ-Management die Nutzung des Beratungsdienstes bzgl. der Sachgebiete monatlich festzustellen. Die Kontrolle der Bearbeitungsdauer wird mit der Kennzahl K3 unterstützt. Um die Spitzenbedarfszeiten der Hotline zu identifizieren und entsprechende Personalplanungen durchführen zu können, wurde die Kennzahl K4 definiert. Einen Gesamtüberblick über die Belastung der Beratung gibt die Kennzahl K7.

Zu der Personengruppe "RZ-Management" zählen wir nicht nur die oberste Führung, sondern auch die Verantwortlichen für den Einkauf und für die Mitarbeiter. Für sie sind die Kennzahlen K12 und K13 bestimmt. K12 kann z.B. bei der Einkaufsplanung verwendet werden, eine Kontrolle der Verantwortlichen für Geräte ermöglicht die Kennzahl K13. Die Kennzahlen K10 und K11 beschreiben die Gesamtsituation im Rechenzentrum bzgl. der Verfügbarkeit und können z.B. zum zwischenbetrieblichen Vergleich eingesetzt werden.

4.2.4 Substruktur "Technische Aspekte des Rechenzentrums"

Zusätzlich zu den Rechenstrukturen und der "Benutzergruppen"-Substruktur implementierten wir auch die im Abschnitt 2.3 vorgestellte Substruktur "Technische Aspekte des Rechenzentrums". Es konnten nur Teile der Gesamtstruktur berücksichtigt werden, für die restlichen Äste der Struktur fehlten uns die Basisdaten in dem Data Warehouse.

Auf der Grundlage der Trouble-Tickets aus ARS und der Ausfälle aus HP OpenView wurden für die Charakteristika "Supportintensität", "Ausfälle", "Nutzung/Auslastung der Beratung" und "Leistungsgrad der Beratung" insgesamt 12 Kennzahlen[35] definiert. Die Einteilung der Kennzahlen in die Ordnungsstruktur "Technische Aspekte des Rechenzentrums" sieht man in der Abbildung 4-5.

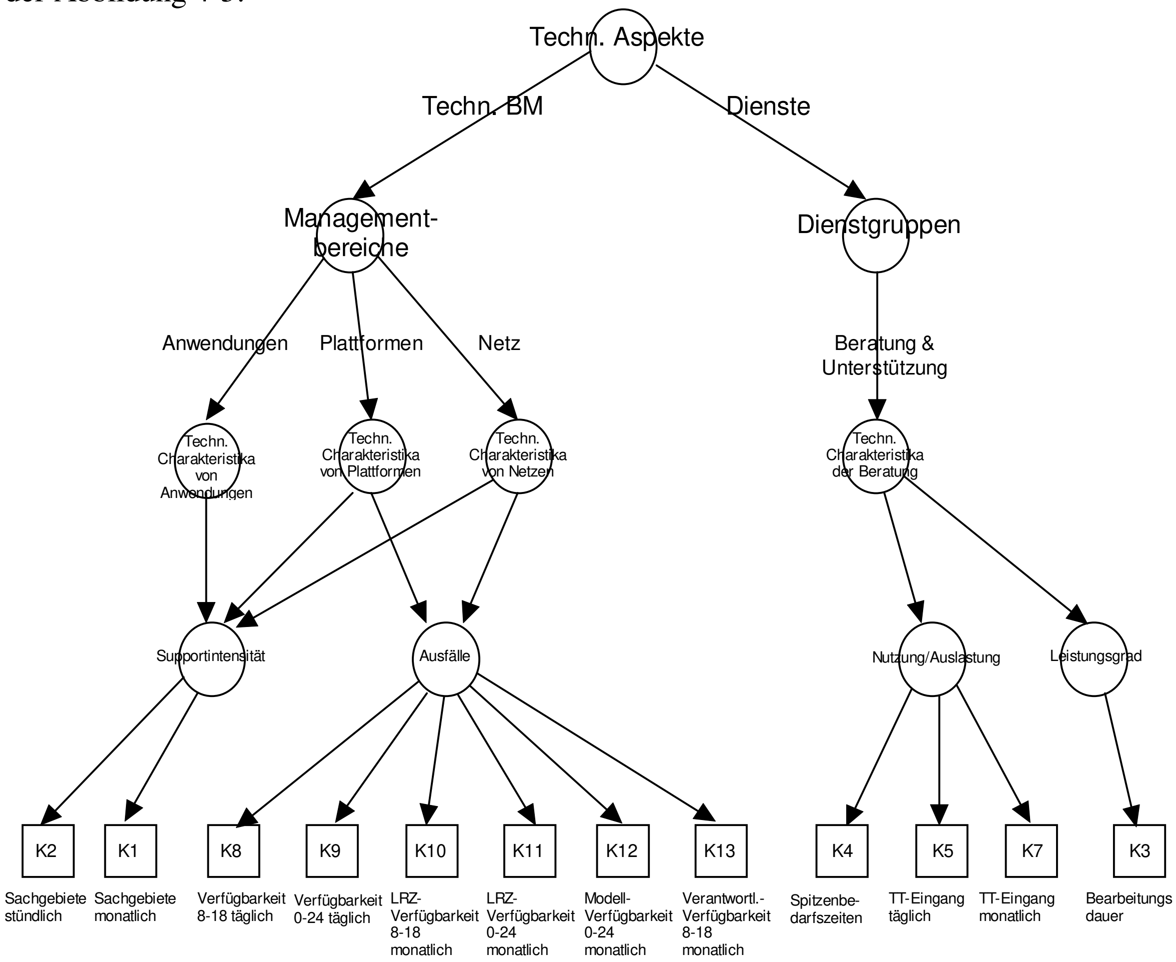


Abbildung 4-5 Substruktur Technische Charakteristika des LRZ

[35] Diese Kennzahlen K1 bis K5 und K7 bis K13 wurden bereits beschrieben.

Die Supportintensität der technischen Betriebsmittel wird durch die zwei Kennzahlen K2 und K1 beschrieben. Über die Zuordnung der Trouble-Tickets zu den Sachgebieten kann man die Anzahl der Probleme bestimmen, die eine bestimmte Geräte- oder Anwendungsklasse[36] verursacht und somit die Intensität des Supports charakterisieren. Leider können keine detaillierteren Angaben gemacht werden, da die Sachgebiete-Hierarchie nicht bis zu den einzelnen Anwendungen, Plattformen und Netzen reicht.

Alle sechs Verfügbarkeitskennzahlen K8 bis K13 werden der Charakteristik "Ausfälle" zugeordnet. Die Kontrolle der Verfügbarkeit kann sowohl monatlich, als auch täglich erfolgen. Außerdem wird zwischen der Arbeitstag-Verfügbarkeit (von 8 bis 18 Uhr) und der 24-Stunden-Verfügbarkeit unterschieden.

Die Kennzahlen K4, K5 und K7 charakterisieren die Nutzung und die Auslastung der LRZ-Dienstleistungsgruppe "Beratung&Unterstützung". K4 besagt, wie die Verteilung der Trouble-Tickets-Erfassungszeiten in einem bestimmten Monat war, und beschreibt somit die Hauptnutzungszeiten der Hotline. Die Kennzahlen K5 und K7 liefern Statistiken über den täglichen bzw. monatlichen Trouble-Ticket-Eingang und sind deshalb Indikatoren für die Steigerung oder Senkung der Nutzung der Beratungsdienste.

Der Leistungsgrad der Beratung kann u.a. über die Kennzahl K3 kontrolliert werden. Diese Größe berechnet die Durchschnitte der Bearbeitungszeiten aller Trouble-Tickets eines bestimmten Sachgebiets in einem Monats und beschreibt somit die Performance der Beratung in diesem Monat. Die Werte dieser Kennzahl sind nur dann aussagekräftig, wenn eine gewisse Stabilität in der Entwicklung beobachtet werden kann. Nur wenn innerhalb einiger Monate ein gleicher oder ein sehr ähnlicher Wert berechnet wird, kann die Kennzahl als Argument verwendet werden. Die zu großen Schwankungen werden nicht durch die kurzfristige Verbesserung oder Verschlechterung des Leistungsfähigkeit verursacht, sondern z.B. durch die besonders niedrigen oder hohen Belastungen der Beratung in diesen Zeitperioden.

4.3 Implementierung des Data-Warehouse-Systems

4.3.1 Datenstrukturen der DW-Datenbank

In der Abbildung 3-4 "Schichtenmodell der DW-Entitäten" haben wir die vorkommenden Entitäten in drei Schichten eingeteilt:

- Wertetabellen und -sichten der Kennzahlen,
- Basisdatentabellen und
- Puffertabellen.

Nur für die Schicht "Basisdaten" läßt sich ein standardisiertes ER-Modell angeben; die Entitäten in beiden anderen Schichten stehen in solchen Beziehungen zueinander, die nicht durch die normalen "ist-ein" und "hat-ein" Relationships des ER-Modells darstellbar sind. Wir müssen deshalb neue Relationship-Typen "aggregiert" und "bezieht-Daten-aus" definieren, um die Beziehungen graphisch darstellen zu können (Abbildung 4-6):

[36] Z.B. "Netzkomponenten->Brouter" oder "Anwendersoftware->Datenbanksysteme".

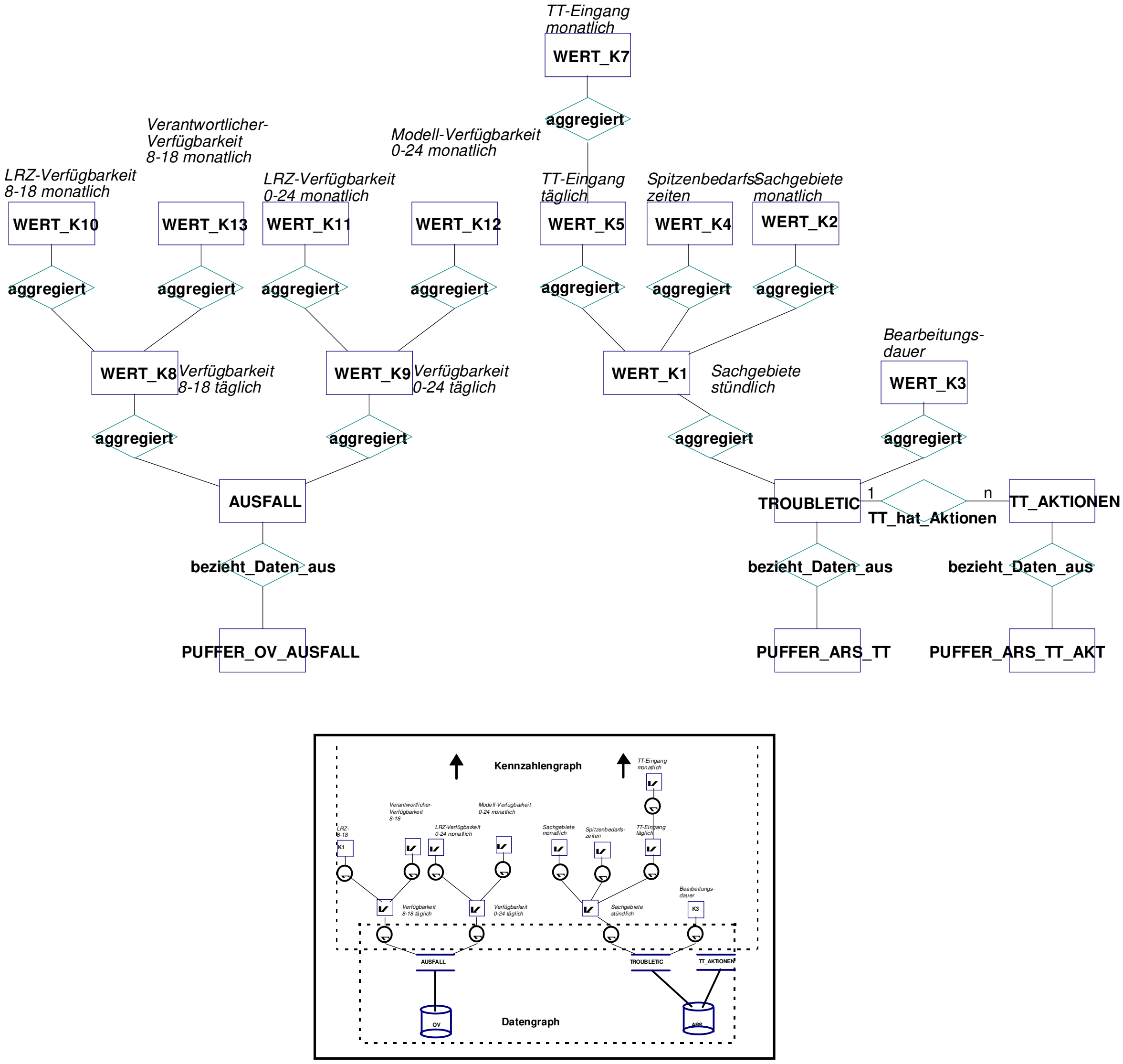


Abbildung 4-6 Datenstrukturen der DW-Datenbank

Das Datenmodell der DW-Datenbank ist sehr eng an die Rechenstrukturen und den Datengraphen des Kennzahlensystems angelehnt. Zum Vergleich haben wir die Abbildung 4-3 “Rechenstrukturen und Datengraph des LRZ-Kennzahlensystems” in die obige Zeichnung integriert.

Die unterste Schicht - die Pufferschicht - nimmt die Daten direkt aus den Datenquellen auf. Wir definierten zwei Puffertabellen für die Datenquelle “ARS” und eine Tabelle für die Datenquelle “OV”. In die Relation “PUFFER_ARS_TT” werden die Hauptinformationen über die einzelnen Trouble-Tickets aus dem AR-System eingestellt. Die Tabelle PUFFER_ARS_TT_AKT dient als Puffer für die “Aktionen” dieser Tickets.

Die nächste Schicht - die Basisdaten-Schicht - besteht wiederum aus drei Tabellen. Diese Tabellen beziehen ihre Daten aus den entsprechenden Puffern. Die Entitäten “TROUBLETIC” und “TT_AKTIONEN” sind in dem ER-Diagramm mit einem Relationship “TT_hat_Aktionen” verbunden - dadurch wird hervorgehoben, daß die Aktionen einem Trouble-Ticket zugeordnet sind.

Oberhalb der Basisdaten-Schicht befinden sich die Wertetabellen und -sichten. Eine Wertetabelle bzw. -sicht aggregiert die Daten aus der zweiten Schicht, sie kann aber auch eine Aggregation von anderen Wertetabellen bzw. -sichten sein. Wir haben in dieser Arbeit ausschließlich Tabellen für die Implementierung der Kennzahlen verwendet.

Die vollständigen Definitionen der Tabellen findet man im Anhang D.

4.3.2 Implementierung des Metadaten-Repository

Wir erwähnten im Abschnitt 3.5 "Entwicklung des Metadaten-Repository", daß für die Modellierung der Metadaten der objektorientierte Ansatz besonders geeignet erscheint. Wir wählten die neueste Entwicklung im Bereich der Modellierungstechniken - die Unified Modelling Language von Booch, Rumbaugh und Jacobson ([UML10]) - aus, um den Design der Metadaten durchzuführen.

Da dem Leser die Elemente dieser Technik wahrscheinlich noch nicht sehr bekannt sind (die Methodik ist erst Anfang 1997 offiziell freigegeben worden), geben wir eine kurze Zusammenfassung der vorkommenden Symbole in der Abbildung 4-7.

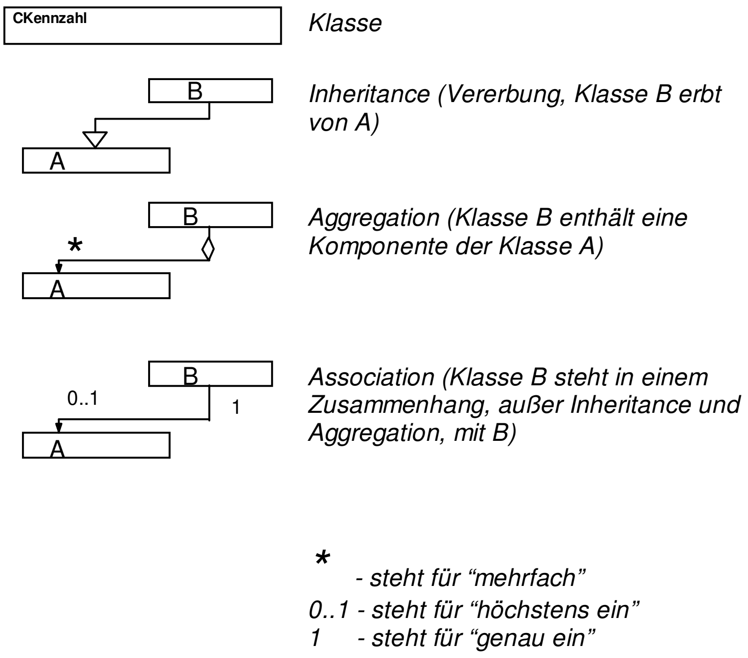


Abbildung 4-7 Symbole der Unified Modelling Language

Es lassen sich folgende Metadaten-Klassen identifizieren[37]:

- CBenutzer (Benutzer-Klasse; relevant für heuristische Strukturen),
- CDiffkrit (Klasse der Differenzierungskriterien; relevant für Ordnungsstrukturen),
- CKennzahl (Kennzahl-Klasse),
- CZusammenhang (Klasse der empirisch-statistischen Zusammanhänge),
- CDiffkritwert (Wert des Differenzierungskriteriums; relevant für Ordnungsstrukturen),
- CKante (Klasse der Kanten in Kennzahlen- und Datengraphen),
- CKnoten (Klasse der Knoten in Kennzahlen- und Datengraphen),
- CVorschrift (Berechnungsvorschrift; relevant für Rechenstrukturen und für den Datengraphen),

[37] Nach einer verbreiteten Konvention beginnen wir den Klassennamen mit einem "C" (**C**lass).

- CDatenquelle (Klasse der Datenquellen),
- CKZGraph (Kennzahlengraph),
- CBDTabelle (Basisdaten-Tabelle) und
- CSubstruktur (Substruktur eines Kennzahlengraphen).

Das UML-Modell der Metadaten ist in der Abbildung 4-8 gezeigt. Die zentrale Klasse in diesem Modell ist der Kennzahlengraph (CKZGraph). Sie enthält eine Komponente der parametrisierten Klasse GRAPH<CKnoten,CKante> aus der Klassenbibliothek LEDA. Zusätzlich zu der GRAPH<,>-Komponente beinhaltet die CKZGraph-Klasse eine Liste von Substrukturen (CSubstrukturen).

Wir entschieden uns für die Verwendung der LEDA-Bibliothek, da in ihr die Funktionalität der gerichteten Graphen und der häufig vorkommenden Graphenalgorithmen vollständig implementiert ist. Außerdem stellt sie eine Fülle von nützlichen Klassen wie Bool, String, List, Hash-Index usw. zur Verfügung.

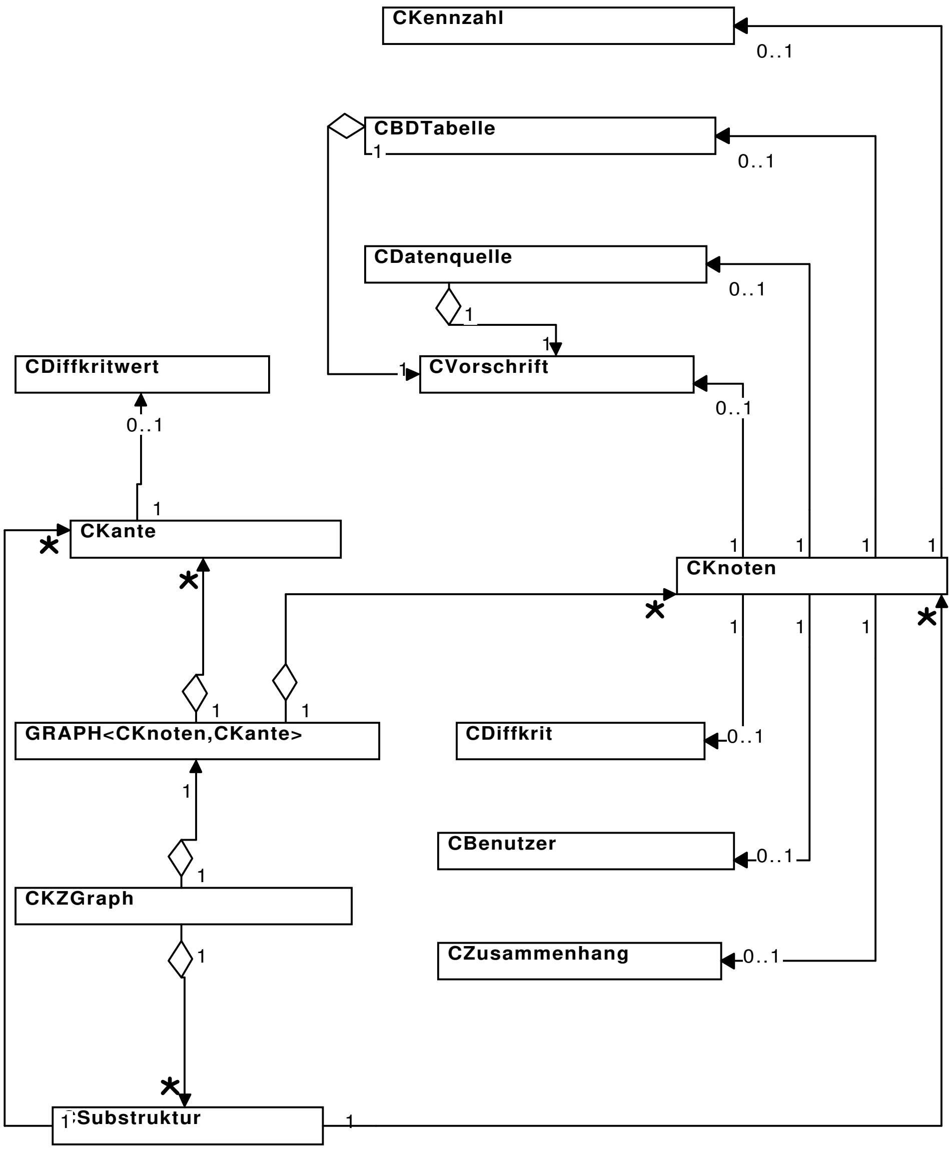


Abbildung 4-8 UML-Modell der Metadaten

Eine Substruktur besteht aus mehreren Knoten und Kanten (eigentlich ein Aggregation-Zusammenhang), aus Implementierungsgründen haben wir aber die Aggregation durch Assoziation ersetzt. Die GRAPH-Komponente beinhaltet bereits die entsprechenden CKante- und CKnoten-Objekte, deshalb würde die Aggregation zu einer Redundanz führen.

Die sieben Knotentypen unseres Kennzahlengraphen werden dadurch berücksichtigt, daß wir acht Metadaten-Klassen definieren. Die CKnoten-Klasse ist die allgemeine Knoten-Klasse, deren Objekte mit entsprechenden "Inhalten" assoziieren. Die Assoziationen zwischen CKnoten und den vier Klassen CVorschrift, CDiffkrit, CBenutzer und CZusammenhang entsprechen den vier Typen der Zusammenhangsknoten in unserem Kennzahlensystem - den Rechenknoten, den Ordnungsknoten, den heuristischen Knoten und den Knoten mit emp.-stat. Zusammenhängen. Der fünfte Assoziationspfeil von CKnoten zu CKennzahl entspricht dem Kennzahlenknoten. Die restlichen zwei Pfeile entsprechen den Datenquellen- und DB-Tabelle-Knoten. Zur Laufzeit führt nur ein Assoziierungspfeil von dem CKnoten-Objekt zu einem existierenden "Inhalt"-Objekt (abhängig von dem Typ des Knotens), deshalb haben die Assoziierungspfeile die Kardinalität "0..1".

Nur ein Kantentyp, nämlich die Kanten in einer Ordnungsstruktur, assoziiert mit anderen Klassen. Der Pfeil von CKante zu CDiffkritwert hat deshalb die Kardinalität "0..1".

Die Aktualisierungsvorschriften der BD-Tabellen (siehe Abschnitt 3.6.4 "Aktualisierung der Basisdatentabellen") werden über eine Aggregation implementiert, d.h. es existiert ein Attribut m_AktVorschrift in der Klasse CBDTabelle. Das gleiche gilt auch für die Extraktions-Vorschriften der Datenquellen (eingeführt im Abschnitt 3.6.3 "Datenanforderung").

In dem Diagramm existieren keine expliziten Relationships zwischen CKnoten und CKante, da sie in der parametrisierten Klasse GRAPH<CKnoten,CKante> "verborgen" sind.

Bei der Implementierung des Data-Warehouse-Systems stand uns nur ein relationales DB-System (Oracle7) zur Abspeicherung der Metadaten zur Verfügung. Wir mußten deshalb einen Bruch machen und das OO-Modell auf das Relationale Modell abbilden. Einen Bruch deshalb, weil unser Data-Migration-System, das intensiv die Metadaten benutzt, vollständig objektorientiert programmiert ist, d.h. in ihm gilt das Modell aus der Abbildung 4-8 weitestgehend. Der erste Schritt des DM-Systems bei der Migration der Daten ist es, die relational abgespeicherte Daten aus der Oracle-DB zu holen und aus ihnen die entsprechenden Objekte zu konstruieren.

Das Ergebnis der Abbildung des OO-Modells auf das Relationale Modell ist im Anhang D zu finden.

4.3.3 Implementierung des Data-Migration-Systems

Die Implementierung des Data-Migration-System erfolgte hauptsächlich in der Programmiersprache C++ mit Embedded SQL. Teile des DM-Systems (alle Aktualisierungsvorschriften von BD-Tabellen und Kennzahlen) wurden mit PL/SQL geschrieben und als Stored Procedures in der Oracle-Datenbank abgelegt (siehe Abbildung 4-9).

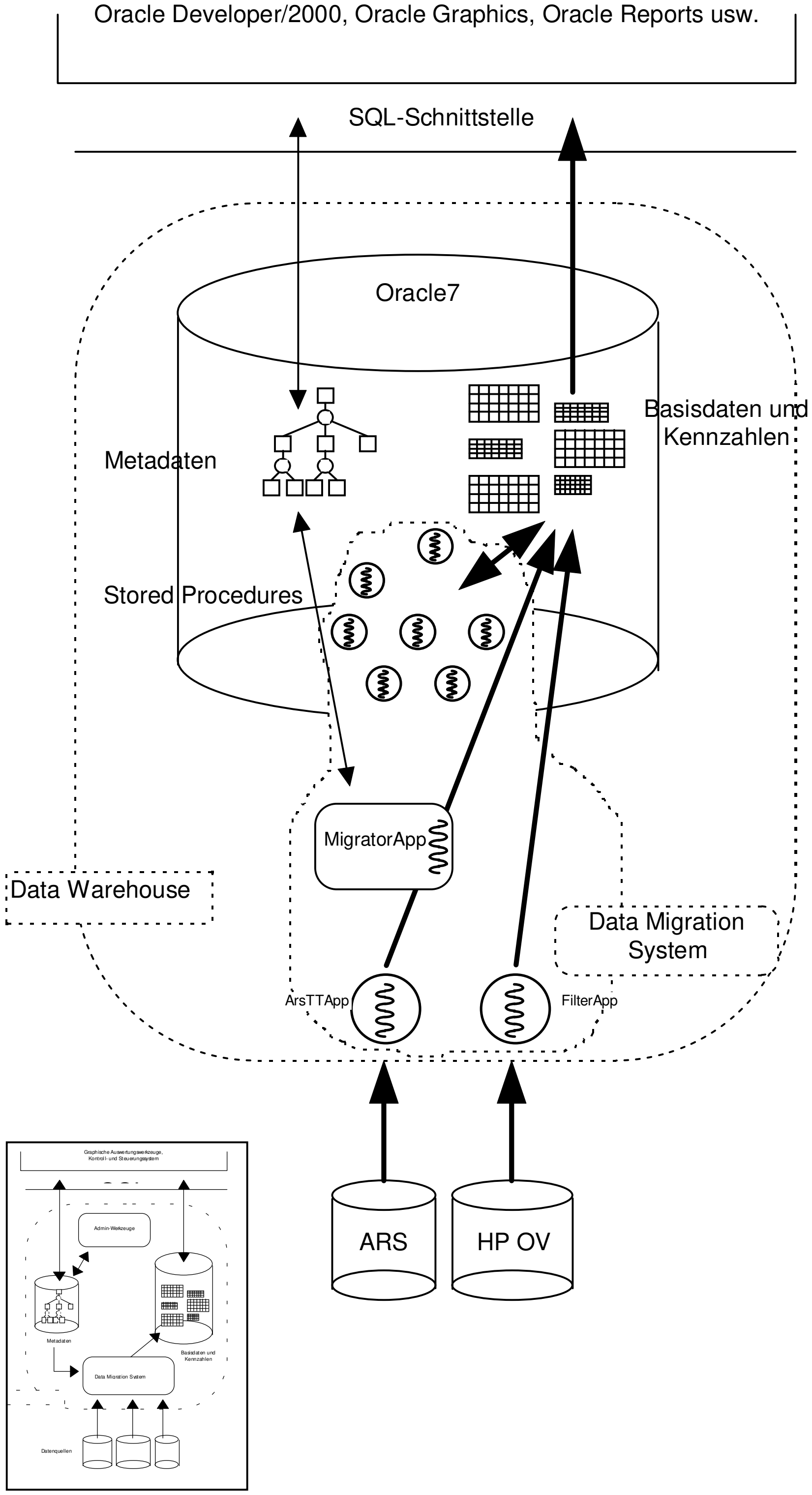


Abbildung 4-9 Kennzahlen-Data-Warehouse im LRZ

Das System besteht aus mehreren Komponenten. Den Kern bildet das Programm "MigratorApp", in dem der Hauptalgorithmus der Datenübernahme implementiert ist. Dieser Algorithmus ist direkt aus der Strukturierten Analyse im Abschnitt 3.6 hergeleitet, so daß wir auf eine nochmalige Erklärung verzichten. Man findet das entsprechende Ablaufdiagramm im Anhang E, Abbildung 5-9.

Die Übernahme der Daten aus der ARS-Datenbank erfolgt mittels eines separaten Programms "ArsTTApp". Dieses Modul entspricht einem "Extraktion_Transformation_Laden"-Bubble aus der Abbildung 3-9 DFD “Datenanforderung”. Es stellt die Trouble-Tickets in die zwei Puffer-Tabellen PUFFER_ARS_TT und PUFFER_ARS_TT_AKT ein.

Im Rahmen eines Fortgeschrittenen-Praktikums am Leibniz-Rechenzentrum ([Merk95]) wurde ein sog. “Filter” gebaut, der die Ausfälle-Traps von der OV-Datenbank abholt und in das ARS als Trouble-Tickets einstellt. Während des Betriebs dieses System mußte man leider feststellen, daß die Anzahl der Ausfälle (und somit auch die Anzahl der Trouble-Tickets) sehr groß war. Deshalb wurde das Modul nicht weiter verwendet.

Wir paßten dieses Programm an unsere Bedürfnisse an, so daß der Filter die Ausfälle nicht mehr als Trouble-Tickets in ARS registriert, sondern sie in das Data Warehouse einstellt. Dieses Programm ist permanent aktiv, es übernimmt in kurzen Intervallen die Daten aus der OV-Datenbank und stellt sie in die Puffer-Tabelle PUFFER_OV_AUSFALL ein. Das Modul hat mit der Steuerungs-Komponente des Systems - dem MigratorApp-Programm - nichts zu tun, es wird mittels *cron* gestartet und über spezifische SNMP-Traps terminiert. Die Änderungen im Programm fielen geringfügig aus, deshalb verzichten wir auf eine genauere Darstellung dieses Moduls und verweisen auf die Original-Beschreibung ([Merk95]).

Die Aufgabe der Programme FilterApp und ArsTTApp ist die Einstellung von operationalen Daten in die Puffer-Tabellen des Data-Warehouse-Systems. Ab dann sind die Stored Procedures für die Weiterverarbeitung der Daten zuständig. Ihr Einsatzgebiet ist die Aktualisierung der Kennzahlen-Wertetabellen und der Basisdatentabellen. Generell existiert pro Kennzahl(-Wertetabelle) und Basisdaten-Tabelle eine Aktualisierungsprozedur.

Die Begründung für den Einsatz von Stored Procedures liegt in der Performance-Steigerung und Reduzierung der Netz- und Datenbank-Belastung. Die Prozeduren werden nämlich von einem nativen Interpretator von Oracle auf der gleichen Plattform wie die Datenbank ausgeführt, so daß die Verarbeitung der Daten “innerhalb” der Datenbank erfolgt. Die Verwendung der Stored Procedures ist nur für die Aktualisierung der Basisdaten- und Wertetabellen sinnvoll (hier wird ausschließlich mit “internen” Daten gearbeitet), für die Datenanforderung bringt ihr Einsatz keine Vorteile, da dort mit externen (aus der Sicht des DBMS Oracle) Daten gearbeitet wird. Außerdem sind die Datenzugriff-API’s der operationalen Systeme meist nur für C/C++-Programme verwendbar, die Stored Procedured werden aber in PL/SQL geschrieben.

Die Applikation “MigratorApp” ist die Steuerungs- und Koordinationskomponente des Data Migrations Systems, sie nimmt an der Verarbeitung der operationalen Daten nur indirekt teil (über die Applikationen FilterApp und ArsTTApp und die Stored Procedures). Ihre Konfiguration bezieht sie aus den Metadaten des Systems. Dort sind nämlich alle wichtigen Informationen, wie die Aufrufe der Programme, Abhängigkeiten zwischen Kennzahlen und Datenquellen usw. persistent abgespeichert.

4.4 Implementierungsergebnisse und Weiterentwicklung des Prototyps

In diesem Kapitel wendeten wir die Methoden des Frameworks zum Aufbau von technischen RZ-Kennzahlensystemen an die Probleme des Leibniz-Rechenzentrums an und implementierten einen Prototyp des LRZ-Kennzahlensystems. Es wurden 12 Kennzahlen aus den Bereichen “Beratung” , “Verfügbarkeit” und “Supportintensität” samt der entsprechenden Kennzahlenstrukturen definiert. Der Aufbau eines Data Warehouses für dieses Kennzahlensystem folgte den Definitionsschritten. Insbesondere wurden zwei operationale Datenquellen in das LRZ-Data-Warehousing integriert und auf der Basis ihrer Daten die Kennzahlen berechnet.

Auf dem Weg von dem Prototyp zu dem vollständigen LRZ-Kennzahlensystem sind folgende Weiterentwicklungen des Systems sinnvoll:

1) Entwicklung einer graphischen Benutzerschnittstelle für die Abfrage der Kennzahlenwerte,
2) Entwicklung von graphischen Werkzeugen zur Unterstützung der Administration des Systems.

Die momentan verfügbare Abfrageschnittstelle besteht aus den Standardwerkzeugen von Oracle (Oracle Graphics, Oracle SqlPlus u.a.). Die Mächtigkeit dieser Tools erlaubt es, ausdrucksvolle Graphiken zu erstellen und die notwendigen Abfragen mittels SQL durchzuführen. Als einzige Verbesserung wäre hier die Integration dieser Werkzeuge in eine dedizierte Software zu nennen. Das Tool könnte z.B. den Kennzahlengraphen graphisch darstellen, eine Navigation in ihm mittels der Maus unterstützen und den Click auf einen Kennzahlenknoten in den Aufruf der entsprechenden Oracle-Graphik umsetzen. Ein ähnliches Tool sollte auch der Administrator des Systems bekommen. Für ihn wäre es nützlich, wenn er die Kennzahlenstrukturen graphisch erstellen und ändern könnte. Dazu könnte z.B. die bereits erwähnte LEDA-Klassenbibliothek verwendet werden, die zusätzlich zu der von uns verwendeten GRAPH-Klasse auch eine GRAPH-Visualisierungsklasse zur Verfügung stellt.

5. Zusammenfassung und Ausblick

Die Hälfte der am LRZ erfaßten Problemmeldungen sind dem Sachgebiet "Netzdienste" zuzuordnen. Seit Anfang 1996 steigt der Anteil der "Netzdienste"-Trouble-Tickets ununterbrochen und beträgt heute mehr als 50% (Anfang 1996 noch 38 %).

Die durchschnittliche Bearbeitungsdauer der Probleme mit dem PC-Benutzerinterface ist im Laufe des Jahres 1996 um mehr als 60 % gesunken und beträgt zur Zeit nur einen Drittel des früheren Wertes (5,5 Kalendertage respektive 15-17 Tage). Die Monatswerte zeichnen sich durch eine relative Stabilität aus, die Schwankungen sind kleiner als 15 %.

Mit diesen Resultaten einer mit Hilfe des LRZ-Kennzahlensystems durchgeführten Kennzahlenanalyse wollen wir den abschließenden Teil unserer Arbeit beginnen.

Das Ergebnis dieser Arbeit sind ein allgemeines Framework zur Definition und Realisierung von technischen RZ-Kennzahlensystemen und ein Prototyp des Kennzahlen-Data-Warehouse für das Leibniz-Rechenzentrum. Diese Arbeit ist weder eine vollständige Untersuchung der theoretischen Aspekte von technischen RZ-Kennzahlensystemen, noch wurde hier ein vollkommenes Kennzahlensystem für das Leibniz-Rechenzentrum entwickelt. In jedem Fall haben wir aber ein weitgehend lückenloses Konzept vorgestellt und an dem Beispiel des LRZ seine Implementierbarkeit bewiesen.

Wegen beschränkter zeitlicher Ressourcen konnten nicht alle Ideen weiterentwickelt oder implementiert werden. Weitere Arbeiten sind auf folgenden Gebieten möglich:
1) Anbindung von neuen Datenquellen,
2) Definition weiterer Kennzahlen und Kennzahlenstrukturen,
3) Durchführung von empirisch-statistischen bzw. Data Mining Untersuchungen.

Die Voraussetzung für die Definition von weiteren Kennzahlen ist die Anbindung von neuen Datenquellen, z.B. der Ascend-Logs, der snmpCollect-Statistiken und des Restes der OV-Datenbank. Je mehr operationale Datenquellen in das Data Warehousing einbezogen werden, desto leistungsfähigere Kennzahlenanalysen könnte man durchführen.

Nachdem in dem Data Warehouse genügend viele Daten gesammelt worden sind, wären weitgehende empirisch-statistische bzw. Data Mining Untersuchungen sinnvoll. Wir vermuten, daß einige interessante Zusammenhänge z.B. im Bereich der Verfügbarkeit von LRZ-Systemen (siehe [DrKa97]) festgestellt werden können. Die Erkenntnisse dieser Untersuchungen könnten auch das Rahmenkonzept bereichern, indem z.B. die allgemeingültigen empirisch-statistischen Zusammenhänge in das Framework integriert würden.

Die von uns aufgestellte Ziel-Mittel-Hierarchie für Rechenzentren könnte verfeinert und auf Praxistauglichkeit validiert werden. Dazu wären grundlegende Überlegungen bzgl. der Ziele und Mittel der Betriebsart "Rechenzentrum" notwendig.

Literaturverzeichnis

[Brau81] Braun, M.: Zielgerichtete Steuerung und systematische Überwachung der betrieblichen Datenverarbeitung. Dissertation TU München, 1981

[Broy92] Broy, M.: Informatik. Eine grundlegende Einführung. Teil 1, Berlin, Heidelberg, Springer Verlag, 1992

[Dieb84] Diebold (Hrsg.): Diebold Kennzahlen-System (DKS). Frankfurt/Main, 1984

[DrKa97] Dreo-Rodosek, G.; Kaiser, T.: Determining the Availability of Distributed Applications, IFIP/IEEE International Symposium on Integrated Management, San Diego, 1997

[Groh79] Grohmann, H.: Kenngrößen zur Steuerung von Rechenzentren. In: DV Aktuell 1979, Schwerpunktthema Steuerung und Überwachung von Rechenzentren, München, Wien 1979, S. 117-133.

[GrTi95] Greiner, M.; Tinhofer G.: Wahrscheinlichkeitstheorie und Statistik für Informatiker. Vorlesungsskriptum, München, 1995. TUM-INFO-03-1995-250/1.-FI

[Hein76] Heinen, E.: Grundlagen betriebswirtschaftlicher Entscheidungen. Das Zielsystem der Unternehmung, 3. Aufl., Wiesbaden, 1976

[LRZ-WWW] LRZ-World Wide Web-Seiten: http://www.lrz-muenchen.de/wir/regelwerk/

[Marc78] De Marco, T.: Structured Analysis and Systems Specification, New York, 1978

[März83] März, T.: Interdependenzen in einem Kennzahlensystem. Dissertation TU München, 1983

[Merk95] Merkl, K.: Implementierung eines Netzverfügbarkeits-Moduls für das Rechnernetz des LRZ. Fortgeschrittenen-Praktikum, Technische Universität München, 1995.

[Meye94] Meyer, C.: Betriebswirtschaftliche Kennzahlen und Kennzahlensysteme. 2. Aufl., Schäffer-Poeschel Verlag, Stuttgart, 1994

[Reic95] Reichmann, T.: Controlling mit Kennzahlen und Managementberichten. 5. Aufl., München, 1995

[SVD80] Schweizerische Vereinigung für Datenverarbeitung (SVD)(Hrsg.): EDV-Kennzahlen. Verlag Paul Haupt, Bern, Stuttgart, 1980

[UML10] Booch, G.; Rumbaugh, J.; Jacobson, I.: Unified Modelling Language for Object-Oriented Development. Documentation Set, Version 1.0, Rational Software Corporation.

[Wies96] Wies, R.; Lohrmann, J.; Picot, A.: Technische Kennzahlen für das IV-Controlling, 1996

Abbildungsverzeichnis

Anhang A Definition der LRZ-Kennzahlen

Die Kennzahlen werden in der Form einer Tabelle präsentiert, unsere Erklärungen zu den Feldern sind kursiv hervorgehoben:

Id	*Hier steht die Funktionalität der Kennzahlenfunktion.*
Einheit	*Hier steht die Einheit der Kennzahlenwerte.*
Verwendung der Kennzahl:	*Hier wird die mögliche Verwendung beschrieben. Insbesondere formulieren wir hier die Frage, auf welche die Kennzahl eine Antwort gibt. Manchmal steht hier nur ein Hinweis darauf, daß diese Kennzahl eine Basis für andere Kennzahlen ist.*
Beispiel:	*Wir geben hier ein Beispiel für einen Kennzahlenwert an und formulieren dann die Aussage in Worten.*
Bemerkung:	*Dieses Feld beinhaltet unsere Bemerkungen zu der Kennzahl.*
Vorschrift:	*Dieses Feld beinhaltet die Berechnungsvorschrift der Kennzahl.*
Wir geben die Vorschriften in einer mathematischer Notation an, obwohl die "echten" Ausdrücke in PL/SQL, einer 4GL-Sprache von Oracle, geschrieben sind. Meistens ist der Übergang von der mathematischen Formel zu der PL/SQL-Anweisung trivial.	

Sachgebiete stündlich

K1	**fct K1** = (Sachgebiet_1: **string**, Sachgebiet_2: **string** , Sachgebiet_3: **string**, Stunde : **date**) **number**
Einheit	Trouble-Tickets
Verwendung der Kennzahl:	Diese Kennzahl beantwortet die Frage: "Wieviele Trouble-Tickets eines bestimmten Sachgebiets wurden in der Stunde H erfaßt ?". Sie kann als Basis für andere Kennzahlen verwendet werden.
Beispiel:	K1("ZS", "Betrieb", "Novell-Server",12/03/97 14:00) = 2 (Es wurden zwei Trouble-Tickets des Sachgebiets "ZS→Betrieb→Novell-Server" zwischen 14 und 15 Uhr am 12. März 1997 erfaßt)
Bemerkung:	Als Granularitätsbasis wurde eine Stunde gewählt, da die in LRZ anfallenden Datenmengen (max. 30 TT pro Tag) eine feinere Unterteilung überflüssig machen. In dem AR-System des LRZ's existieren maximal 3 Stufen der Sachgebiete-Hierarchie. Wir speichern diese Stufen in drei unterschiedlichen Feldern ab: Sachgebiet_1, Sachgebiet_2 und Sachgebiet_3. Diese Kennzahl dient als Basis für andere Kennzahlen. So ist es z.B. möglich, auf ihrer Basis die Kennzahlen K4 "Spitzenbedarfzeiten" , K5

	“TT-Eingang täglich” usw. zu definieren. Diese abgeleiteten Kennzahlen besitzen eine höhere Aussagekraft, als die Basiskennzahl K1. Der Vorteil dieser abgestuften Definition liegt in der Performance-Steigerung, da die abgeleiteten Kennzahlen die Ergebnisse der Vorberechnung verwenden können.
Vorschrift:	

$$K1(s1, s2, s3, stunde) = \left|\left\{TROUBLETIC: erfasst \in stunde \wedge sachgebiet1 = s1 \wedge sachgebiet2 = s2 \wedge sachgebiet3 = s3\right\}\right|$$

(Anzahl der Einträge der Basisdatentabelle TROUBLETIC, für deren Attribute die Bedingung gilt)

Sachgebiete monatlich

K2	**fct K2** = (Sachgebiet_1: **string**, Sachgebiet_2: **string** , Sachgebiet_3: **string**, Monat : **date**) **number**
Einheit	Trouble-Tickets
Verwendung der Kennzahl:	Diese Kennzahl beantwortet die Frage: “Wieviele Trouble-Tickets eines bestimmten Sachgebiets wurden in dem Monat ‘Monat’ erfaßt ?”.
Beispiel:	K2(“ZS”,”Betrieb”,”Novell-Server”,03/97) = 15 (Im März 1997 wurden 15 Trouble-Tickets des Sachgebiets “ZS→Betrieb→Novell-Server” erfaßt)
Bemerkung:	Diese Kennzahl ist eine Aggregation der Kennzahl K1.
Vorschrift:	

$$K2(s1, s2, s3, monat) = \sum_{stunde \in monat} K1(s1, s2, s3, stunde)$$

∅ Bearbeitungsdauer

K3	**fct K3** = (Sachgebiet_1: **string**, Sachgebiet_2: **string** , Sachgebiet_3: **string**, Monat : **string**) **float**
Einheit	Tage / Trouble-Ticket
Verwendung der Kennzahl:	Diese Kennzahl beantwortet die Frage: “Wie lange werden Trouble-Tickets eines Sachgebiets durchschnittlich bearbeitet?”. Sie sagt nichts darüber aus, wieviele Manntage die RZ-Mitarbeiter für die Lösung des Problems aufgewendet haben (im LRZ existiert keine problembezogene Arbeitszeiterfassung). Somit kann sie nicht zur Kostenrechnung herangezogen werden. Der Aussagewert der Kennzahl liegt z.B. in der Vorhersage des Lösungszeitpunkts der Probleme. Außerdem können besonders “schwere” oder “leichte” Probleme identifiziert werden.
Beispiel:	K3(“ZS”,”Betrieb”,”Novell-Server”,03/97) = 0.67 (Ein Novell-Server-Problem wurde in März im Durchschnitt 16 Stunden bzw. 0.67 Tage lang bearbeitet)

Bemerkung:	Es wird ein Durchschnitt über die in einem bestimmten Monat eingegangenen Trouble-Tickets gebildet. Trouble-Tickets mit dem aktuellen Status “Erfasst”, “In Bearbeitung” oder “Fehler gefunden” werden als nicht abgeschlossen betrachtet.
Vorschrift:	

$$K3(s1, s2, s3, monat) = \frac{\sum_{TT \in M} (TT.geschlossen - TT.erfasst)}{|M|}, \text{ wobei}$$

$$M = \{ TT \in TROUBLETIC \mid TT.erfasst \in monat \wedge TT.geschlossen \neq NULL \wedge TT.sachgebiet1 = s1 \wedge TT.sachgebiet2 = s2 \wedge TT.sachgebiet3 = s3\}$$

Spitzenbedarfszeiten

K4	**fct K4** = (S : **number,** Monat**: date**) **number**
Einheit	Trouble-Tickets / Std.
Verwendung der Kennzahl:	Diese Kennzahl beantwortet die Frage : “Wie viele Trouble-Tickets werden in der S-ten Stunde eines Tages durchschnittlich erfaßt?”.
Beispiel:	K4(14, 04/97) = 2 (In April 1997 wurden in der Zeit zwischen 14:00 und 15:00 durchschnittlich zwei TT erfaßt)
Bemerkung:	Zum Ausgleich der möglichen Schwankungen wird ein Durchschnitt über einen Monat gebildet. Ein engerer Zeitraum (eine Woche oder ein Tag) kommt wegen der kleinen Datenmengen (max. 30 TT pro Tag) nicht in Frage. Längere Zeitintervalle (Quartal oder Jahr) lassen sich leicht auf die Monatsdaten abstützen. Zeiträume, die sich nicht an die üblichen Teilungen der Zeit (Stunde, Tag, Woche, Monat, Quartal, Jahr) halten, sind wenig sinnvoll.
Vorschrift:	

$$K4(S, monat) = \frac{1}{20} \times \sum_{stunde \in monat, stunde \sim S} K1(*,*,*, stunde)$$

(Die Kennzahl K1 wird über alle Sachgebiete aggregiert, und zwar nur für Stunden, die im Monat “monat” liegen und S “entsprechen”. Als durchschnittliche Anzahl von Arbeitstagen in einem Monat wurden 20 Tage zugrundegelegt)

TT-Eingang täglich

K5	**fct K5** = (Tag : **date**) **number**
Einheit	Trouble-Tickets
Verwendung der Kennzahl:	Diese Kennzahl beantwortet die Frage : “Wieviele Trouble-Tickets wurden an einem bestimmten Tag erfaßt?”.
Beispiel:	K5(12/03/97) = 25 (Am 12. März 1997 wurden 25 Tickets erfaßt)
Bemerkung:	

Vorschrift:	

$$K5(tag) = \sum_{stunde \in tag} K1(*,*,*,stunde)$$

TT-Eingang monatlich

K7	**fct K7** = (Monat : **date**) **number**
Einheit	Trouble-Tickets
Verwendung der Kennzahl:	Diese Kennzahl beantwortet die Frage : "Wieviele Trouble-Tickets wurden in einem bestimmten Monat erfaßt?".
Beispiel:	K7(03/97) = 280 (In März 1997 wurden 280 Trouble-Tickets erfaßt)
Bemerkung:	
Vorschrift:	

$$K7(monat) = \sum_{tag \in monat} K5(tag)$$

Verfügbarkeit 8-18 täglich

K8	**fct K8** = (Hostname: **string**, Portadresse: **string** , Standort: **string**, Hersteller : **string**, Modell: **string,** Verantwortlicher: **string,** Tag**: date**) **number**
Einheit	
Verwendung der Kennzahl:	Diese Kennzahl beantwortet die Frage: "Wie hoch war die Verfügbarkeit eines Hosts (eines Ports des Hosts) an einem bestimmten Tag zwischen 8 und 18 Uhr?". Sie kann als Basis für andere Kennzahlen verwendet werden.
Beispiel:	K8("sun1",…,12/03/97) = 0.98 (Die 8-18 Verfügbarkeit von sun1 betrug am 12. März 1997 98 %)
Bemerkung:	Die Parameter Standort, Hersteller, Modell und Verantwortlicher sind nicht unabhängig von der Parameterkombination (Hostname,Portadresse). Sie wurden aus Implementierungsgründen in die Kennzahl integriert (um K13 und andere Kennzahlen auf ihrer Basis berechnen zu können)
Vorschrift:	

$$K8(hn, p, s, hs, m, v, tag) = 1 - \frac{summe\_ausfallzeiten\_8\_18(hn, p, tag)}{1\,Tag}$$

(summe_ausfallzeiten_8_18(hostname, portadresse, tag) steht für eine Vorschrift, die die Ausfallzeiten eines Ports an einem bestimmten Tag zwischen 8 und 18 Uhr berechnet. Der Wert wird in Bruchteilen eines Tages geliefert)

Verfügbarkeit 0-24 täglich

K9	**fct K9** = (Hostname: **string**, Portadresse: **string** , Standort: **string**, Hersteller : **string**, Modell: **string,** Verantwortlicher: **string,** Tag**: date**) **number**
Einheit	
Verwendung der Kennzahl:	Diese Kennzahl beantwortet die Frage: “Wie hoch war die Verfügbarkeit eines Hosts (eines Ports des Hosts) an einem bestimmten Tag zwischen 0 und 24 Uhr?”. Sie kann als Basis für andere Kennzahlen verwendet werden.
Beispiel:	K9(“sun1”,…,12/03/97) = 0.95 (Die 0-24 Verfügbarkeit von sun1 betrug am 12. März 1997 95 %)
Bemerkung:	siehe Bemerkung zu K8
Vorschrift:	$$K9(hn,p,s,hs,m,v,tag) = 1 - \frac{summe\_ausfallzeiten\_0\_24(hn,p,tag)}{1\ Tag}$$ (summe_ausfallzeiten_0_24(hostname, portadresse, tag) steht für eine Vorschrift, die die Ausfallzeiten eines Ports an einem bestimmten Tag zwischen 0 und 24 Uhr berechnet. Der Wert wird in Bruchteilen eines Tages geliefert)

LRZ-Verfügbarkeit 8-18 monatlich

K10	**fct K10** = (Monat : **date**) **number**
Einheit	-
Verwendung der Kennzahl:	Diese Kennzahl beantwortet die Frage: “Wie hoch war die Gesamt-LRZ-Verfügbarkeit in einem bestimmten Monat (zwischen 8 und 18 Uhr) ?”.
Beispiel:	K10(03/97) = 99,5 % (In März 1997 war die LRZ-Verfügbarkeit zwischen 8 und 18 Uhr 99,5 Prozent)
Bemerkung:	In unserer prototypischen Implementierung dieser Verfügbarkeits-Kennzahl werden alle Hosts und Ports gleichbehandelt, ein zentraler Brouter ist also gleich gewichtet wie eine Workstation. Eine sinnvolle Erweiterung wäre die Einführung von Gewichten für verschiedene Geräteklassen.
Vorschrift:	$$K10(monat) = \underset{tag \in monat}{\varnothing} K8(*,*,*,*,*,*,tag)$$ (Durchschnitt über alle K8-Werte mit …)

LRZ-Verfügbarkeit 0-24 monatlich

K11	**fct K11** = (Monat : **date**) **number**

Einheit	-
Verwendung der Kennzahl:	Diese Kennzahl beantwortet die Frage: "Wie hoch war die Gesamt-LRZ-Verfügbarkeit in einem bestimmten Monat (zwischen 0 und 24 Uhr) ?".
Beispiel:	K11(03/97) = 99 % (In März 1997 betrug die LRZ-Verfügbarkeit zwischen 0 und 24 Uhr 99 Prozent)
Bemerkung:	siehe Bemerkung zu K10
Vorschrift:	

$$K11(monat) = \underset{tag \in monat}{\varnothing} K9(*,*,*,*,*,*,tag)$$

(Durchschnitt über alle K9-Werte mit …)

Modell-Verfügbarkeit 0-24 monatlich

K12	**fct K12** = (Modell : **string**, Monat : **date**) **number**
Einheit	-
Verwendung der Kennzahl:	Diese Kennzahl beantwortet die Frage: "Wie hoch war die Verfügbarkeit von Geräten eines bestimmten Modells in einem Monat (zwischen 0 und 24 Uhr) ?".
Beispiel:	K11("SPARCstation 20",03/97) = 99 % (In März 1997 betrug die Verfügbarkeit der Sparc-Stations 99 Prozent)
Bemerkung:	
Vorschrift:	

$$K12(mdl, monat) = \underset{tag \in monat}{\varnothing} K9(*,*,*,*,mdl,*,tag)$$

(Durchschnitt über alle K9-Werte mit …)

Verantwortlicher-Verfügbarkeit 8-18 monatlich

K13	**fct K13** = (Verantwortlicher: **string**, Monat : **date**) **number**
Einheit	-
Verwendung der Kennzahl:	Diese Kennzahl beantwortet die Frage: "Wie hoch war die Verfügbarkeit der Geräte, für die eine bestimmte Person verantwortlich ist, in einem Monat (zwischen 8 und 18 Uhr)?".
Beispiel:	K13("Mayer",03/97) = 100 % (In März 1997 war die Verfügbarkeit der Geräte, für die der Mitarbeiter Mayer verantwortlich ist, 100 Prozent)
Bemerkung:	Diese Kennzahl berücksichtigt die Ausfälle der Geräte nur in der Zeit, wo die verantwortliche Person auch unmittelbar eingreifen kann. Wegen des direkten Personenbezugs dieser Kennzahl, müßte u.U. der

	Betriebsrat über ihre Verwendung informiert werden.
Vorschrift:	

$$K13(verantwortlicher, monat) = \underset{tag \in monat}{\varnothing} K8(*,*,*,*,*, verantwortlicher, tag)$$

(Durchschnitt über alle K8-Werte mit …)

Anhang B Beispiel einer Kennzahlenanalyse

Dieser Teil der Arbeit soll einen Einblick in die Möglichkeiten der Kennzahlenanalyse geben, die unser System bereits im prototypischen Zustand ermöglicht. Als Analyseobjekte haben wir die Beratungskennzahlen ausgewählt, d.h. die Größen K1 bis K7.

Wir beginnen die Analyse mit der Betrachtung der Kennzahl K7 "TT-Eingang monatlich", die einen guten Überblick der Nutzung der Beratungsdienste gibt. In ihrer Entwicklung ist kein Trend zu beobachten, weder eine Senkung noch ein Anstieg der Nutzung der Beratungsdienste (Abbildung 5-1).

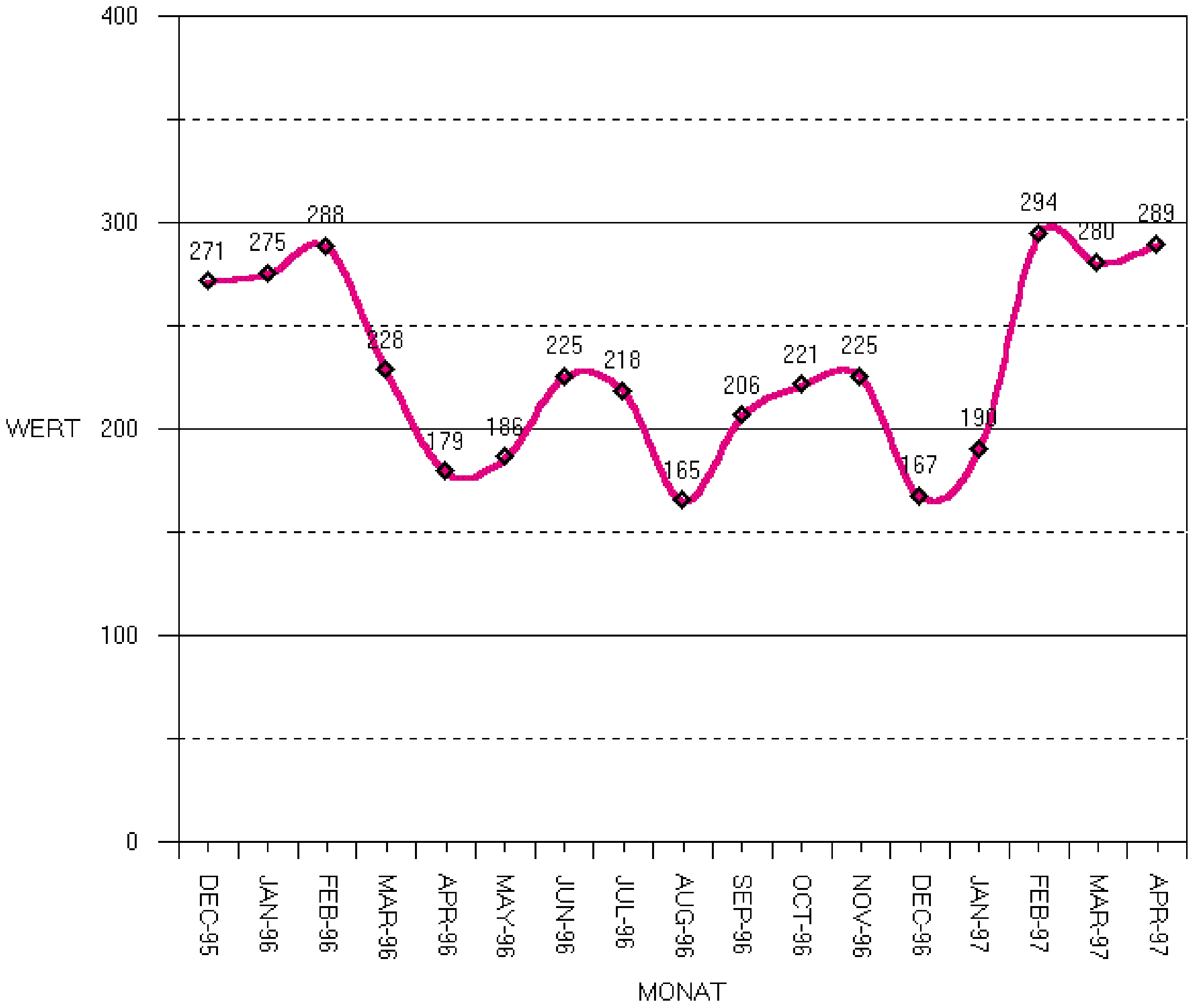


Abbildung 5-1 Beispiel der Kennzahl K7 (TT-Eingang monatlich)

Zwar existieren zum Teil sehr große saisonale Unterschiede in der monatlichen Erfassung der Trouble-Tickets, jedoch pendeln die Werte in der Regel um die 220-230 TT/Monat Marke.

Wenn man sich die Struktur der Problemmeldungen näher anschaut, so fällt einem die Dominanz der Netzprobleme mit über 46,7%[38] auf (Abbildung 5-2). Die zwei anderen problematischen Sachgebiete - die Zentralen Systeme und das Benutzerinterface PC - kommen zusammen nicht einmal auf die Hälfte der Netz-Tickets.

[38] Dieser Wert bezieht sich auf alle Trouble-Tickets in dem Data Warehouse.

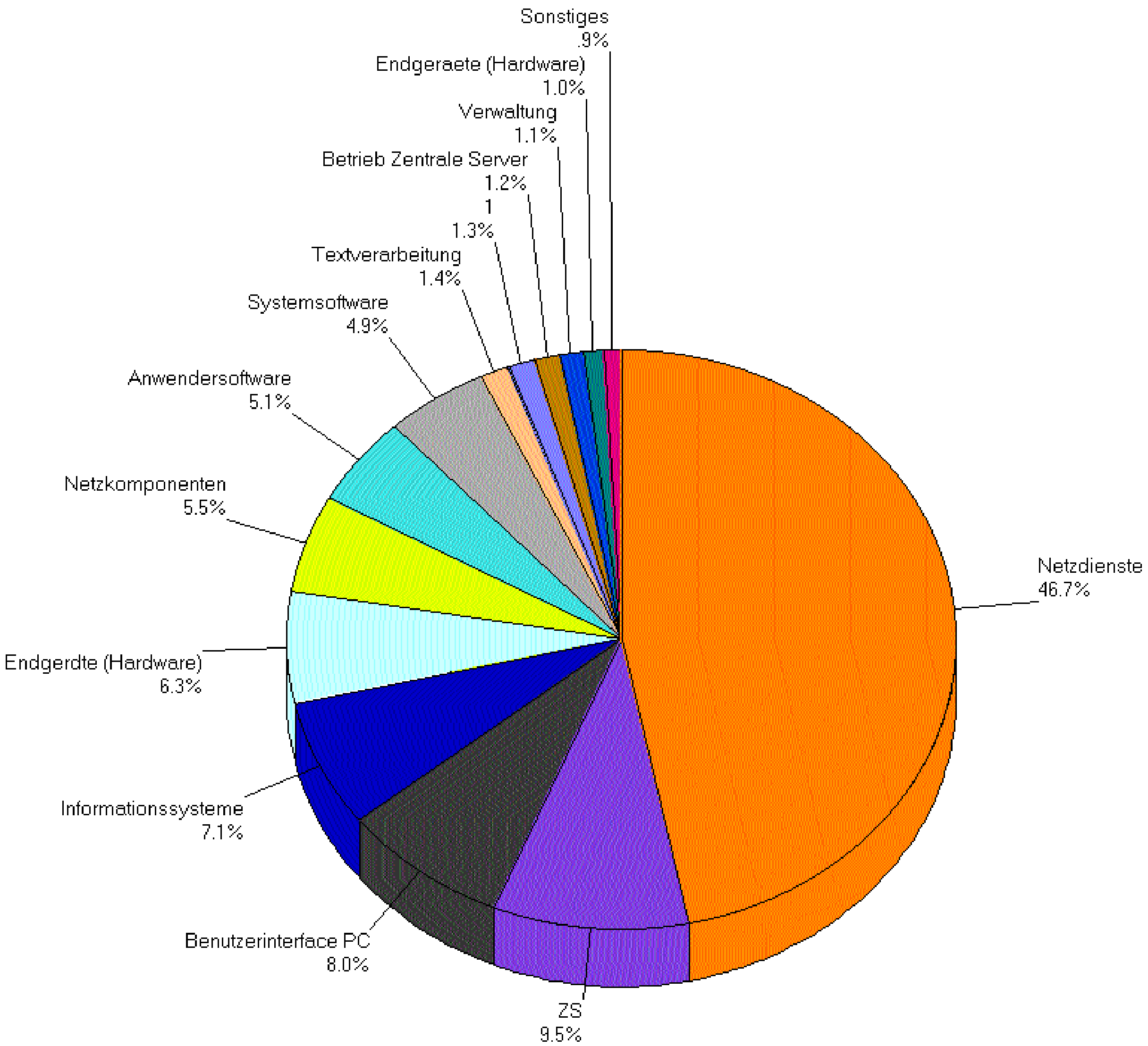


Abbildung 5-2 TT-Verteilung nach Sachgebieten

Die Entwicklung des Anteils der Netzdienste-Trouble-Tickets an der Gesamtzahl der Problemmeldungen spricht eine noch deutlichere Sprache (Abbildung 5-3). Der Anteil wuchs seit Anfang 1996 langsam aber stetig von 39% in April 1996 auf 51% in April 1997.

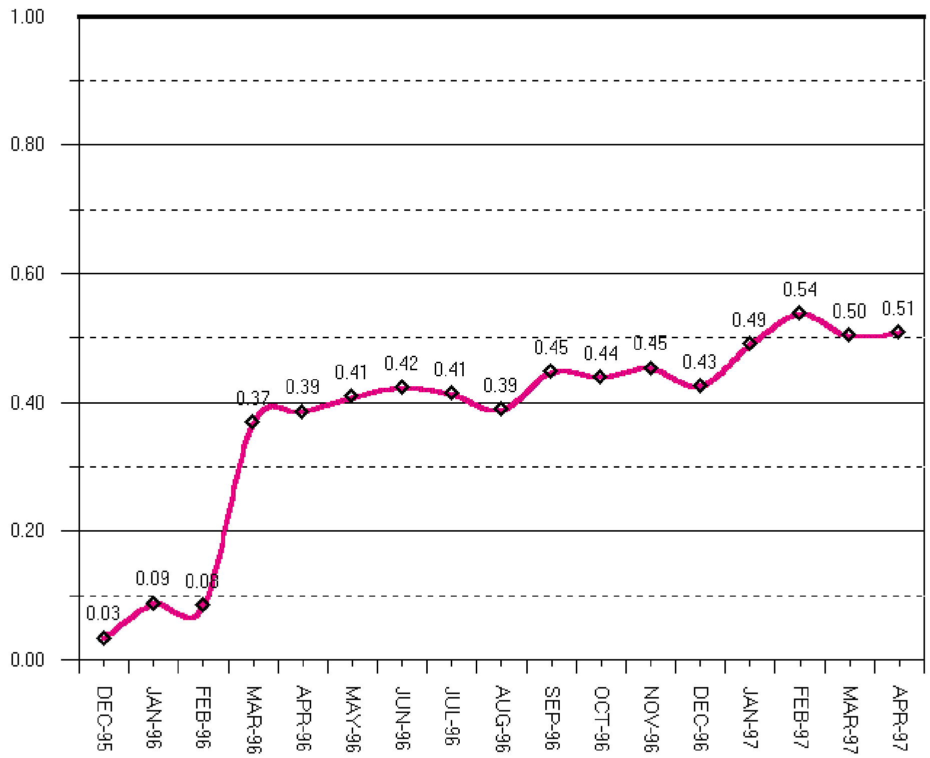


Abbildung 5-3 Entwicklung des Anteils der Netzdienste-Trouble-Tickets

Die Tendenz der Benutzerinterface-Probleme (Abbildung 5-4) entspricht weitgehend der Gesamttendenz (Abbildung 5-1) - im August 1996 hatten beide Kurven ihre Minima, im Februar 1997 - ihre Maxima. Die Monate März 1997 und Dezember 1997 bestätigen die starke Korrelation, die restlichen Monate sind weder als Bestätigung noch als Widerlegung der Hypothese einzustufen. Da der Anteil der PC-Tickets zu niedrig ist, um die Gesamttendenz so stark zu beeinflussen, kann die Annahme gemacht werden, daß die Gesamtentwicklung die treibende Kraft für die PC-Probleme ist. Alternativ könnte auch eine dritte Größe[39] die Ursache für beide Entwicklungen sein, so daß keine direkte Verbindung zwischen den PC-Problemen und der Gesamtproblemmenge zu existieren braucht. Die Überprüfung dieser Hypothesen ist nur mit einer tieferen Analyse möglich.

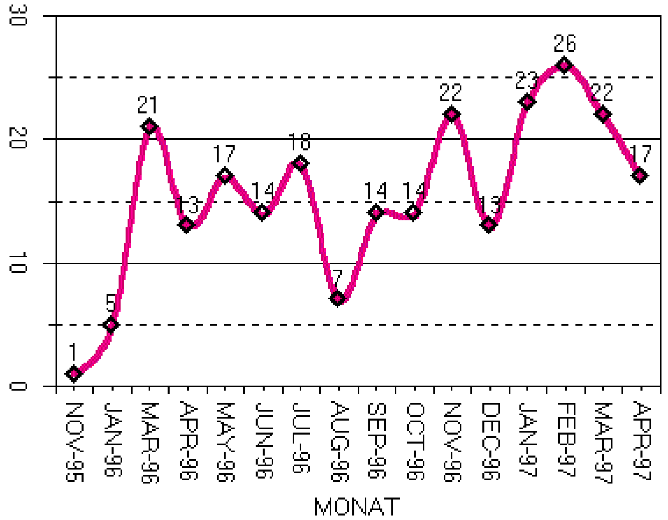


Abbildung 5-4 Beispiel der Kennzahl K2 ("Benutzerinterface PC"-Trouble-Tickets)

[39] Eine solche Größe könnte z.B. die Anzahl der aktiven Benutzer sein - Probleme werden von den Benutzern gemeldet, deshalb kann der Anstieg oder die Senkung der Anzahl der aktiven Benutzer die entsprechende Entwicklung bei der Problemmeldung verursachen.

Die durchschnittlichen Bearbeitungszeiten der Probleme mit dem Benutzerinterface PC fielen von 15-17 Tagen in der ersten Hälfte 1996 auf weniger als 6 Tage am Ende 1996/Anfang 1997 (Abbildung 5-5). Dabei kann man eine fast konstante Bearbeitungszeit von ca. 5,5 Tagen an der Jahresschwelle 1996/1997 beobachten. Die “Ausreißer” August 1996, Oktober 1996 und April 1997 haben wahrscheinlich deshalb so niedrige Werte (ca. 1,5 Tage), weil in diesen Monaten die Anzahl der Probleme sehr gering war und die Verantwortlichen sich intensiver um die einzelnen Tickets kümmern konnten.

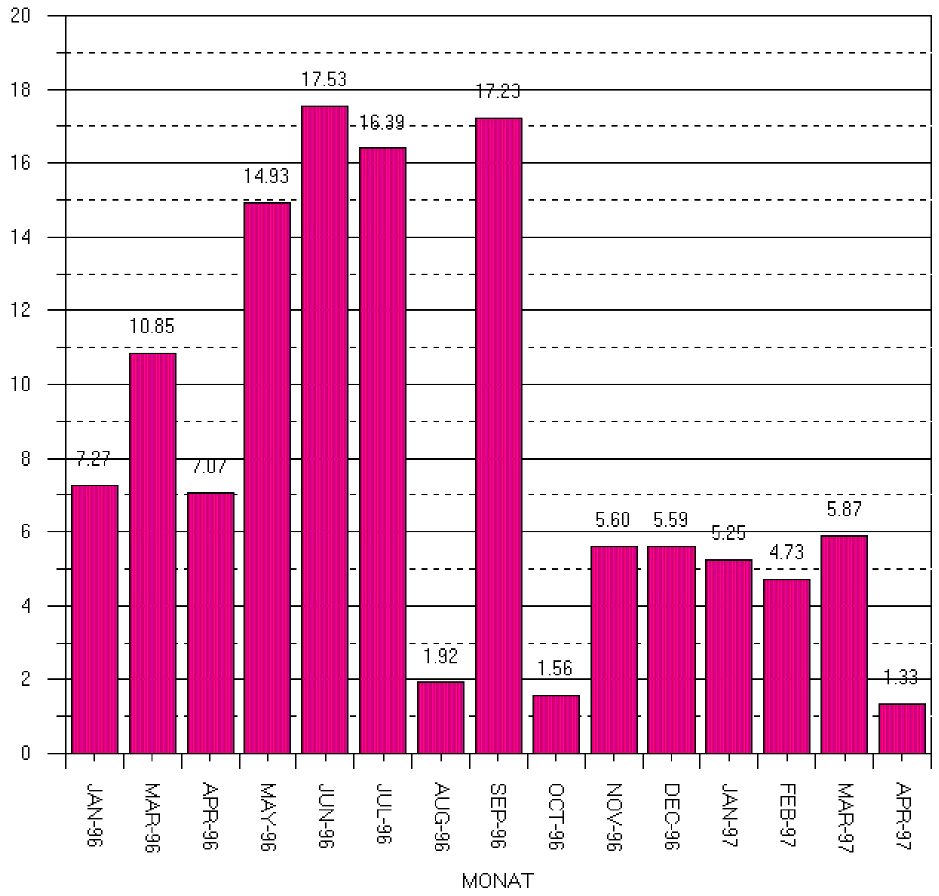


Abbildung 5-5 Beispiel Kennzahl K3 (Bearbeitungszeiten eines Benutzerinterface-Problems)

Die Verteilung der Erfassungszeiten auf die einzelnen Stunden eines Arbeitstages im März 1997 sieht man in der Abbildung 5-6. Generell gilt, daß pro Stunde normalerweise 2 Tickets erfaßt werden, maximal aber 3. Der Groß der Registrierung fällt auf die Zeiten 10-12 und 14-15 Uhr. Davor, danach und dazwischen ist ein relativ “ruhiger” Betrieb.

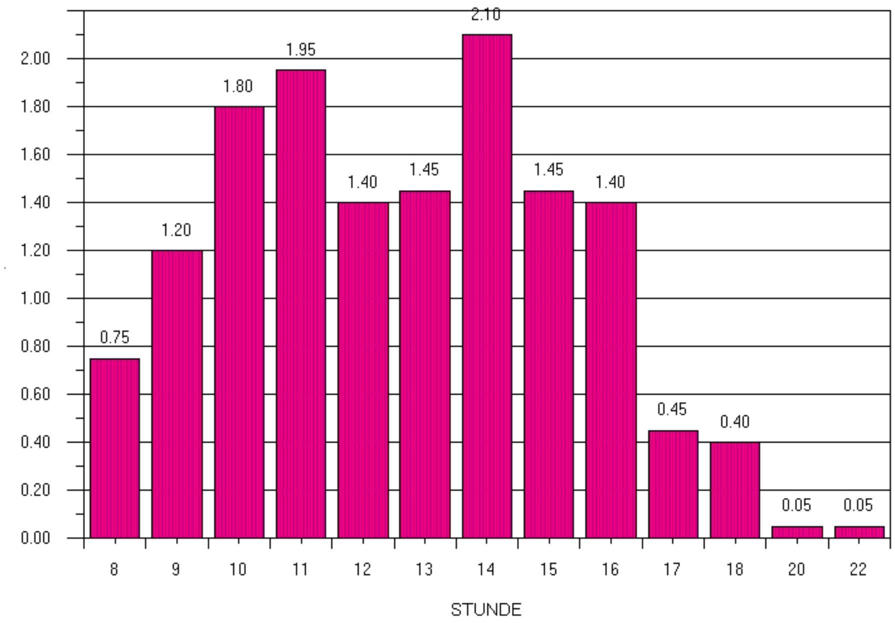


Abbildung 5-6 Beispiel Kennzahl K4 (Tageszeiten der Problemmeldung)

Die Untersuchung der Monatsverteilung der eingehenden Trouble Tickets bestimmt die Wochentage Dienstag und Mittwoch als die beliebtesten Beschwerdetage (Abbildung 5-7; Analysezeitraum März 1997). Die Problemmeldung am Freitag ist dagegen 2,5 mal geringer, als in der Wochenmitte. Auch den Montag nutzen die Anwender kaum für die Inanspruchnahme der LRZ-Beratung.

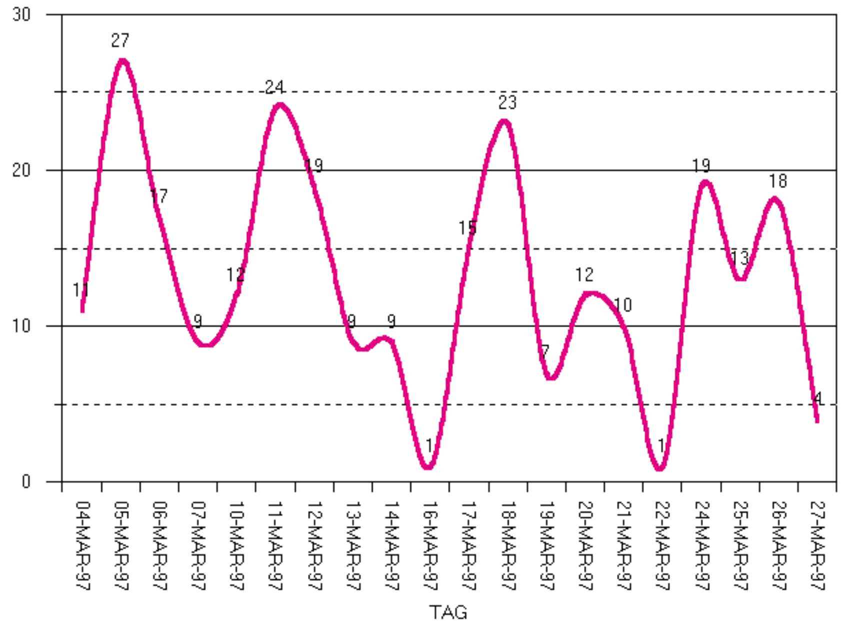


Abbildung 5-7 Beispiel Kennzahl K5 (Monatliche Eingangsstatistik für März 1997)

Anhang C Operationale Datenquellen im LRZ

C.1 ARS

ARS.Trouble-Ticket-Neu		**Lokation**	ARS-Datenbank auf dbserver.lrz-muenchen.de
Quelltyp	“Schema” der ARS-Datenbank		
Abgespeicherte Daten	Trouble-Tickets der LRZ- Hotline und -Beratung		
Format			
Größe	ca. 8000 Einträge (Stand März 1997)	**Zuwachsraten**	ca. 20 (max. 30) Einträge täglich
Extraktionsmethode	API des AR-Systems		
Bemerkungen	Ein AR-Schema ist mit einer Tabelle in einem relationalen DBMS vergleichbar (das Schema ist, wie auch eine Tabelle, in Felder geteilt). Eine Besonderheit dieser Schematas ist, daß in den Feldern auch Listen abgespeichert sein können.		
Beispieldaten			
"TT0007295","01/16/97 10:25:24","Geschlossen","Vorname Name"," Vorname Name ","01/16/97 14:51:00","Frau","?????",,"????","Gering",,"Netzdienste->Mail (Sun,HP,PC)",," Vorname Name ","MailProblem unter Pegasus (Waehlmodem)","1.Kennung ******* 2.PegasusMail (Waehlmodem) 3.Grosse Mail geladen 4.kein Loeschen der vorhanden Mail moeglich 5.kein Empfang von neuer Mail",,,,"01/16/97 10:25:24 Vorname Name Neuer Verantwortlicher: Vorname Name 01/16/97 14:51:00 Vorname Name auf Anweisung von Frau Vorname Name uebergrosse Mail geloescht ","nein",,," Vorname Name "," Vorname Name ",,,,," Vorname Name "," Erfasst Vorname Name 01/16/97 10:25:24 In Bearbeitung Ausgesetzt Fehler gefunden Geschlossen Vorname Name 01/16/97 14:51:00 "			

Feld -Nr.	Feldname	Kommentar	Übernehmen?
1	TT-Nummer,	OK	Ja
2	Erfassungsdatum,	OK, das Feld ist aber mit “Status-History” redundant	Nein
3	Zustand,	OK	Ja
4	Bearbeiter /in,	OK	Ja
5	Erfasser/in,	OK	Ja
6	Bearbeitungsdatum,	OK, das Feld ist aber redundant mit “Aktionen”	Ja (aus Performance-Gründen bei Abfragen)
7	Benutzer/in,	OK	Ja
8	Institut,	Nicht OK, da keine einheitliche Abkürzungen für Institute. Z.B. • Anorg. Chem. Inst. TU, • Anorg. Chemie, • Anorganische Chemie, • Anorganische Chemie TU Außerdem: für viele TT leer	Nein
9	Tel./Fax,	Nicht OK, da in 30% der Fälle leer, kein einheitliches Format,	Nein

		z.B.: • (089) 67 25 30 • +49 (8091) 6022 • 0 81 61 / 71 33 01 • -28715	
10	Email,	Nicht OK, da • oft leer, • kein einheitliches Format	Nein
11	Dringlichkeit,	OK	Ja
12	siehe TT,	OK (aber zwei unterschiedliche Formate z.B. 1234 und TT0001234)	Ja
13	Sachgebiet,	OK	Ja, geteilt in drei Felder
14	Suchen in Hilfeliste,	Hilfsfeld	Nein
15	Verantwortlich,	OK	Ja
16	Kurzbeschr.,	Nicht OK, da Fließtext	Nein
17	Beschreibung,	Nicht OK, da Fließtext	Nein
18	Standort,	Nicht OK, da • meistens leer (in 85% der Datensätze) • keine einheitlichen Kürzel	Nein
19	Schlagworte,	Nicht OK, da Fließtext	Nein
20	Benutzer erbittet Antwort?	OK, aber ohne “Benutzer benachrichtigt” nicht verwendbar (siehe Bemerkung zum Feld “Benutzer/in benachrichtigt?”)	Nein
21	Aktionen,	OK	Ja, als eine eigene Relation
22	Benutzer/in benachrichtigt?	Nicht OK, da ständig falsch ausgefüllt : Kontakt mit dem Benutzer vorhanden (ersichtlich aus den Kommentaren zu Aktionen), jedoch Feldwert = nein	Nein
23	Gruppenbenachrichtigung?	Nicht OK, da immer leer	Nein
24	Soll geschlossen werden am	Nicht OK, da immer leer (100% der Fälle)	Nein
25	Mitarbeiter2	Hilfsfeld	Nein
26	Mitarbeiter1,	Hilfsfeld	Nein
27	Email?,	Hilfsfeld	Nein
28	Message,	Hilfsfeld	Nein
29	Dummy,	Hilfsfeld	Nein
30	Eskalation,	Hilfsfeld	Nein
31	Vorgesetzter,	OK	Ja
32	Status-History	OK	Ja, geteilt in 5 Felder

C.2 snmpCollect

SnmpCollect sammelt zur Zeit die Werte von folgenden MIB-Variablen:

- IPXForward.0,
- Slot%CpuUsed.1 … .9,
- avgBusy1,
- avgBusy5,
- callStatusHighWaterMark,
- ifInErrors.1 … .18 (1-18 durchgängig),
- ifInOctets.1 … .602 (1-602 hierarchisch),
- ifOutErrors.1 … .18 (1-18 durchgängig),
- ifOutOctets.1 … .602 (1-602 hierarchisch),
- ipForwarding.0,
- ipInReceives,
- snmpInPkts

Es sind folgende Informationen über Variablen verfügbar:

- Datum_1,
- Domain-Name des Agenten ,
- Datum_2 in sec.,
- Datum_3 in sec.,
- IP-Addresse

Als Beispiel eines Variablen-Logs beschreiben wir die Datei ifInOctets.10.

<table>
<tr><td colspan="2">ifInOctets.10</td><td>Lokation</td><td>sunmanager:/usr/OV/databases/snmpCollect/if InOctets.10</td></tr>
<tr><td>Quelltyp</td><td colspan="3">Flat File</td></tr>
<tr><td>Abgespeicherte Daten</td><td colspan="3">Werte der MIB-Variable ifInOctets.10 von ca. 6 LRZ-Broutern. Daten ab 1.1.95.</td></tr>
<tr><td>Format</td><td colspan="3">Binärdatei, pro Eintrag folgende Felder:
Datum_1,
DNS-Name des Agenten ,
Datum_2 in sec.,
Datum_3 in sec.,
IP-Adresse des Agenten</td></tr>
<tr><td>Größe</td><td>ca. 100K Einträge (Stand 15.3.97), Eintragsgröße ca. 100 Byte</td><td>Zuwachsraten</td><td>ca. 250 Einträge täglich</td></tr>
<tr><td>Extraktionsmethode</td><td colspan="3">Export der Daten aus der Binärdatei in ASCII-Format mittels snmpColDump.</td></tr>
<tr><td>Bemerkungen</td><td colspan="3">Das Polling-Intervall ist in 100% der Einträge genau 30 Minuten lang (1800 sec.). Für manche Kombinationen von Intervallen und Agenten fehlen allerdings die Werte.</td></tr>
<tr><td>Beispieldaten</td><td colspan="3"></td></tr>
<tr><td colspan="4">03/16/97 18:48:54 dfvgate 0 858532734 858534534 129.187.10.25
03/16/97 18:48:54 bro1cz.lrz-muenchen.de 124184 858532734 858534534 129.187.10.254
03/16/97 18:48:56 bro0q1.camp-wst.lrz-muenchen.de 36.4301 858532736 858534536 141.40.1.254
03/16/97 19:18:39 bro2bz.lrz-muenchen.de 102.085 858534519 858536319 129.187.4.245</td></tr>
</table>

03/16/97 19:18:41	bro2cz.lrz-muenchen.de	75350.7	858534521	858536321	129.187.4.254

Feld - Nr	Feldname	Kommentar	Über- nehmen ?
1	Datum_1	OK, jedoch Zweck nicht völlig klar. Der Wert des Feldes liegt zwischen den Werten der Felder Datum_2 und Datum_3 (ungefähr in der Mitte des Zeitintervalls : Mittelpunkt +/- 3-4 Min.).	Unklar (entweder nur Datum_1 oder nur Datum_2+Datum_3)
2	Domain-Name des Agenten	OK. Bemerkung: es sind fast ausschließlich Brouter präsent. Das Feld ist "redundant" mit der IP-Adresse.	Unklar (Entweder DNS oder IP)
3	Wert der Variablen	OK, aber : oft Werte des Typs FLOAT. Unklar ist der Sinn solcher Werte für die Variable ifInOctets.	Ja
4	Datum_2	OK. Angabe in Sekunden seit 1.1.1970. Wahrscheinlich Anfang des Gültigkeitsintervalls für den Wert der MIB-Variable.	Unklar
5	Datum_3	OK. Angabe in Sekungen seit 1.1.1970. Wahrscheinlich das Ende des Gültigkeitsintervalls für den Wert der MIB-Variable. Das Zeitintervall zw. Datum_3 und Datum_2 ist stets 30 Min.	Unklar
6	IP-Adresse	OK	Unklar

C.3 Trap-Logs

Trapd.log	**Lokation**	sunmanager:/mgmtdata/OV/share/log/trapd.log	
Quelltyp	Flat File		
Abgespeicherte Daten	SNMP-Traps des gesamten LRZ-Netzes (incl. TU, LMU)		
Format	ASCII-Datei, nur 1 Eintragstyp, 17 Felder, kein einheitliches Feldtrennzeichen (gleichzeitig werden Leerzeichen, Tab-Zeichen und Semikolon benutzt).		
Größe	Variable Größe der Datei, da permanente Löschungen. Die Größe der Einträge ist auch variabel (abhängig von dem Trap-Typ). Durchschnittlich 100 Byte.	**Zuwachsraten**	ca. 10.000 neue Einträge täglich, abhängig von dem Wochentag
Extraktionsmethode	Extraktion von neuen Einträgen am Ende der Datei. Berücksichtigung des Datums der letzten Extraktion.		
Bemerkungen			
Beispieldaten			

```
835254869 1 Thu Jun 20 09:14:29 1996 hpsiegert3.informatik.tu-muenchen.de  N IF lan0 up;1 17.1 6 58916866 28216
835254869 1 Thu Jun 20 09:14:29 1996 hpsiegert3.informatik.tu-muenchen.de  M Node up;1 17.1 6 58916864 28216
835254914 1 Thu Jun 20 09:15:14 1996 hpljrbg1.informatik.tu-muenchen.de    N IF HP up;1 17.1 6 58916866 62045
```

Feld-Nr.	Feldname	Kommentar	Übernehmen?
1	Trap-Datum	OK, Angabe in sec. seit 1.1.1970	
2	Zweck unklar	Nicht OK, Werte : 1, 2, 3, 4, 5, 6, 7, 8. In 80 % der Einträge ist der Wert gleich 1.	

3	Wochentag	OK, aber redundant mit Feld 1	
4	Monat	OK, aber redundant mit Feld 1	
5	Tag	OK, aber redundant mit Feld 1	
6	Stunde	OK, aber redundant mit Feld 1	
7	Minute	OK, aber redundant mit Feld 1	
8	Sekunde	OK, aber redundant mit Feld 1	
9	Jahr	OK, aber redundant mit Feld 1	
10	Name des SNMP-Agenten	Nicht OK: Oft Angabe der IP-Adresse anstatt des DNS-Namens. In vielen Fällen werden IP-Segmente angegeben (z.B. 131.159 oder 129.187.1). Manchmal werden <none>-Werte geschrieben.	
11	Zweck unklar	Nicht OK. Werte: -, c, D, d, E, F, I, M, N, o, P, R, s, T, t.	
12	Trap-Info	Nicht OK. Kein einheitliches Format. Mehr als 40 Typen der Traps. Das Feld muß geparst werden, um Informationen zu erhalten.	
13	Zweck unklar	Nicht OK. Werte : 1,2,3,4,5. In 75% der Fälle - 1, in 20% - 2	
14	Zweck unklar	Nicht OK. Werte : 17.1, 03.02.02, .1.3.6.1.4.1.298.2.1.1. In 99,9% der Fälle Wert = 17.1	
15	Zweck unklar	Nicht OK. Werte : 0,4, 6. Wert 6 tritt dann und nur dann auf, wenn Feld 14 = 17.1 ist.	
16	Zweck unklar	Nicht OK.	
17	Zweck unklar	Nicht OK.	

C.4 Wählzugangsstatistiken

Detail		**Lokation**	sunmanager:/usr/local/adm/radacct/ascend3.lrz-muenchen.de/detail<JJMMTT>
Quelltyp	Flat File		
Abgespeicherte Daten	ASCII-Datei, 2 Einträge pro Anruf. 2 Eintragstypen : Informationen zum Beginn des Anrufs (Acct-Status-Type = Start) und Informationen zum Ende des Anrufs (Acct-Status-Type = Stop). Der letzte Eintragstyp beinhaltet alle Informationen des Ersten.		
Format	ASCII-Datei, verschiedene Typen von Einträgen, den Feldwerten werden die Feldnamen vorangestellt		
Größe	Eine Datei pro Woche. Die Daten sind über den Zeitraum 9.12.96 bis heute verfügbar. Die Datei der letzten Woche ist nicht komprimiert.	**Zuwachsraten**	ca. 20 MB oder 50K Einträge (beiden Typs) pro Woche.
Extraktionsmethode			
Bemerkungen			
Beispieldaten			

```
Mon Mar  3 02:25:10 1997
	User-Name = ***********
	NAS-Identifier = 129.187.24.125
	NAS-Port = 20110
	Acct-Status-Type = Stop
	Acct-Delay-Time = 0
	Acct-Session-Id = "35787"
```

```
    Acct-Authentic = RADIUS
    Acct-Session-Time = 173
    Acct-Input-Octets = 7971
    Acct-Output-Octets = 99033
    Acct-Input-Packets = 201
    Acct-Output-Packets = 192
    Ascend-Disconnect-Cause = pppRcvTerminate
    Ascend-Connect-Progress = prLanSessionUp
    Ascend-Data-Rate = 28800
    Ascend-PreSession-Time = 14
    Ascend-Pre-Input-Octets = 497
    Ascend-Pre-Output-Octets = 470
    Ascend-Pre-Input-Packets = 6
    Ascend-Pre-Output-Packets = 12
    Ascend-First-Dest = 129.187.255.255
    Client-Port-DNIS = "2881010"
    Framed-Protocol = PPP
    Framed-Address = 129.187.24.124

Mon Mar  3 02:28:50 1997
    User-Name = *********
    NAS-Identifier = 129.187.24.125
    NAS-Port = 20222
    Acct-Status-Type = Start
    Acct-Delay-Time = 0
    Acct-Session-Id = "35791"
    Acct-Authentic = RADIUS
    Client-Port-DNIS = "2881010"
    Framed-Protocol = PPP
    Framed-Address = 129.187.24.66
```

Ascend.log		**Lokation**	sunmanager:/usr/local/adm/ascend.log
Quelltyp	Flat File		
Abgespeicherte Daten			
Format			
Größe	Eine Datei pro Woche. Die Dateien der letzten 5 Wochen sind in einem komprimierten Zustand vorhanden (ca. 50 MB, z.B. ascend.log.0.gz, ascend.log.1.gz usw.). Die Datei der aktuellen Woche ist nicht komprimiert.	**Zuwachsraten**	ca. 40 MB (nicht komprimiert) pro Woche. Wöchentlich ca. 500K Einträge (pro Anruf mehrere Einträge)
Extraktionsmethode			
Bemerkungen	ca. 25K Anrufe pro Woche		
Beispieldaten			

```
Mar  9 10:14:53 ascend1 ASCEND: call 71 AN slot 8 port 6 VOICE 288101
Mar  9 10:14:56 ascend2 ASCEND: slot 0 port 0, line 2, channel 14, Incoming Call, 81789102
Mar  9 10:14:56 ascend2 ASCEND: slot 9 port 1, Assigned to port, 81789102
Mar  9 10:14:56 ascend3 ASCEND: slot 0 port 0, line 1, channel 27, Incoming Call, MBID 086
Mar  9 10:14:56 ascend3 ASCEND: slot 7 port 7, Assigned to port, MBID 086
Mar  9 10:14:57 ascend1 ASCEND: slot 9 port 5, LAN session up, lmu10870@studlmu
Mar  9 10:14:58 ascend2 ASCEND: slot 9 port 1, line 2, channel 14, Call Connected, 81789102
Mar  9 10:14:58 ascend2 ASCEND: call 61 AN slot 9 port 1 64K 2881190
Mar  9 10:14:58 ascend3 ASCEND: call 128 AN slot 7 port 7 VOICE 2881010
```

Anhang D Definitionen der DW-Datenstrukturen

PUFFER_OV_AUSFALL	**Beschreibung**	Puffer-Tabelle

DDL-Definition

```
create table puffer_ov_ausfall(
        pufrec_id           number primary key,
        hostname            varchar2(255),
        portadresse         char(15),
        standort            varchar2(255),
        hersteller          varchar2(255),
        modell              varchar2(255),
        verantwortlicher    varchar2(60),
        ausfall_beginn char(19),
        ausfall_ende        char(19)
);
```

PUFFER_ARS_TT	**Beschreibung**	Puffer-Tabelle

DDL-Definition

```
create table puffer_ars_tt(
        pufrec_id               number primary key,
        tt_nummer               char(9),
        zustand                 varchar2(15),
        bearbeiterin            varchar2(30),
        erfasserin              varchar2(30),
        bearbeit_datum_str      char(19),
        bearbeit_datum_num      number,
        benutzerin              varchar2(255),
        dringlichkeit           varchar2(8),
        siehe_tt                char(9),
        sachgebiet1             varchar2(64),
        sachgebiet2             varchar2(64),
        sachgebiet3             varchar2(64),
        verantwortlich          varchar2(30),
        vorgesetzter            varchar2(30),
        erfasst                 char(19),
        in_bearbeitung          char(19),
        fehler_gefunden         char(19),
        geschlossen             char(19),
        ausgesetzt              char(19)
);
```

PUFFER_ARS_TT_AKT	**Beschreibung**	Puffer-Tabelle

DDL-Definition

```
create table puffer_ars_tt_akt(
        pufrec_id           number primary key,
        tt_nummer           char(9),
        ttakt_datum         char(19),
```

```
	ttakt_bearbeiterin	varchar2(30)
);
```

AUSFALL	**Beschreibung**	Basisdatentabelle

DDL-Definition

```
create table ausfall(
	ausfall_id		number primary key,
	hostname		varchar2(255),
	portadresse		char(15),
	standort		varchar2(255),
	hersteller		varchar2(255),
	modell			varchar2(255),
	verantwortlicher	varchar2(60),
	ausfall_beginn	date,
	ausfall_ende		date
);
```

TROUBLETIC	**Beschreibung**	Basisdatentabelle

DDL-Definition

```
create table troubletic(
	tt_nummer		char(9)			primary key,
	zustand			varchar2(15)		not null,
	bearbeiterin		varchar2(30),
	erfasserin		varchar2(30),
	bearbeit_datum	date,
	benutzerin		varchar2(255),
	dringlichkeit		varchar2(8),
	siehe_tt		char(9),
	sachgebiet1		varchar2(64)		not null,
	sachgebiet2		varchar2(64),
	sachgebiet3		varchar2(64),
	verantwortlich	varchar2(30),
	vorgesetzter		varchar2(30),
	erfasst			date			not null,
	in_bearbeitung	date,
	fehler_gefunden	date,
	geschlossen		date,
	ausgesetzt		date
);
```

TT_AKTIONEN	**Beschreibung**	Basisdatentabelle

DDL-Definition

```
create table tt_aktionen(
	ttakt_id		number primary key,
	tt_nummer		references troubletic(tt_nummer),
	ttakt_datum		date,
	ttakt_bearbeiterin	varchar2(30)
);
```

SUBSTRUKTUR	**Beschreibung**	Metadatentabelle

DDL-Definition

```
create table substruktur(
        sub_id      char(8)         primary key,
        sub_name    varchar2(255)   not null,
        sub_typ     char(1)         check(sub_typ in ('O','R','S','H'))
);
```

KNOTEN	**Beschreibung**	Metadatentabelle

DDL-Definition

```
create table knoten (
        kn_id           char(8) primary key,
        kn_typ          char(1) not null,
        kn_name         varchar2(255) not null,
        sub_id          references substruktur,
        kn_inhalt_id    char(8) not null,
        check (kn_typ in ('O', 'S', 'K', 'R', 'H', 'B', 'D'))
);
```

KANTE	**Beschreibung**	Metadatentabelle

DDL-Definition

```
create table kante(
        ka_id           char(8) primary key,
        ka_typ          varchar2(8),
        u_kn_id         references knoten,
        z_kn_id         references knoten,
        sub_id          references substruktur,
        ka_inhalt_id    char(8),
        check(ka_typ in ('EH','ES','ER1','ER2','ER3','EO1','EO2','EB'))
);
```

KENNZAHL	**Beschreibung**	Metadatentabelle

DDL-Definition

```
create table kennzahl(
        kz_id               char(8) primary key,
        kz_name             varchar2(255),
        kz_wt_name          char(18) not null,
        kz_funktionalitaet varchar2(255)
);
```

DIFFKRIT	**Beschreibung**	Metadatentabelle

DDL-Definition

```
create table diffkrit(
        dk_id       char(8) primary key,
        dk_name     varchar2(255) not null
);
```

BENUTZER	**Beschreibung**	Metadatentabelle

DDL-Definition

```
create table benutzer(
        benutzer_idchar(8) primary key,
        benutzer_info  varchar2(255) not null
);
```

ZUSAMMENHANG	**Beschreibung**	Metadatentabelle

DDL-Definition

```
create table zusammenhang(
        zus_id        char(8) primary key,
        signifikanz  float not null,
        kor_koeffizient    float not null
);
```

DIFFKRITWERT	**Beschreibung**	Metadatentabelle

DDL-Definition

```
create table diffkritwert(
        dkw_id       char(8) primary key,
        dkw_wert   varchar2(255) not null
);
```

VORSCHRIFT	**Beschreibung**	Metadatentabelle

DDL-Definition

```
create table vorschrift(
        vt_id     char(8) primary key,
        vt_minispec    varchar2(255),
        vt_typ        char(8) check(vt_typ in ('STORPROC','EXTNPROG')),
        vt_aufruf    varchar2(255) not null
);
```

DATENQUELLE	**Beschreibung**	Metadatentabelle

DDL-Definition

```
create table datenquelle(
        dq_id        char(8) primary key,
        dq_info      varchar2(255),
        etl_vt_idreferences vorschrift
);
```

BD_TABELLE	**Beschreibung**	Metadatentabelle

DDL-Definition

```
create table bd_tabelle(
        bdt_id       char(8) primary key,
        bdt_name   char(18) not null,
        akt_vt_id   references vorschrift
```

```
);
```

WERT_K1	**Beschreibung**	Wertetabelle

DDL-Definition

```
create table wert_k1(
        wert_id         number          primary key,
        sachgebiet1     varchar2(64)    not null,
        sachgebiet2     varchar2(64),
        sachgebiet3     varchar2(64),
        stunde          date            not null,
        wert            number          not null,
        unique (sachgebiet1,sachgebiet2,sachgebiet3,stunde)
);
```

WERT_K2	**Beschreibung**	Wertetabelle

DDL-Definition

```
create table wert_k2(
        wert_id         number          primary key,
        sachgebiet1     varchar2(64)    not null,
        sachgebiet2     varchar2(64),
        sachgebiet3     varchar2(64),
        monat           date            not null,
        wert            number          not null,
        unique (sachgebiet1,sachgebiet2,sachgebiet3,monat)
);
```

WERT_K3	**Beschreibung**	Wertetabelle

DDL-Definition

```
create table wert_k3(
        wert_id         number          primary key,
        sachgebiet1     varchar2(64)    not null,
        sachgebiet2     varchar2(64),
        sachgebiet3     varchar2(64),
        monat           date            not null,
        wert            number          not null,
        unique (sachgebiet1,sachgebiet2,sachgebiet3,monat)
);
```

WERT_K4	**Beschreibung**	Wertetabelle

DDL-Definition

```
create table wert_k4(
        wert_id         number          primary key,
        stunde          number          check(stunde between 0 and 23),
        monat           date            not null,
        wert            number          not null,
        unique (stunde,monat)
);
```

WERT_K5	**Beschreibung**	Wertetabelle

DDL-Definition

```
create table wert_k5(
	wert_id		number		primary key,
	tag		date		not null,
	wert		number		not null,
	unique (tag)
);
```

WERT_K7	**Beschreibung**	Wertetabelle

DDL-Definition

```
create table wert_k7(
	wert_id		number		primary key,
	monat		date		not null,
	wert		number		not null,
	unique (monat)
);
```

Anhang E Data-Migration-Algorithmus

Der Algorithmus der Data Migration wurde in einer eigenen Klasse “CMigrator” untergebracht. CMigrator erbt, wie auch alle anderen Klassen in unserem System, die mit der Oracle-Datenbank zu tun haben, von einen Hilfsklasse “CEsqlExecutor”. Diese Klasse implementiert die übliche Funktionalität der Fehlerbehandlung eines ESQL-“Ausführers”.

CMigrator hat zwei Attribute (members) : m_Datenbank und m_KZGraph. Das erste Attribut - ein Objekt der Klasse CDatenbank - ist für die Transaktions-Funktionalität des Migrators verantwortlich (die Klasse CDatenbank implementiert die Operationen Connect, Commit, Rollback und Disconnect), das zweite Attribut ist der Kennzahlengraph, der bei der Migration als die Quelle für alle notwendigen Metadaten dient.

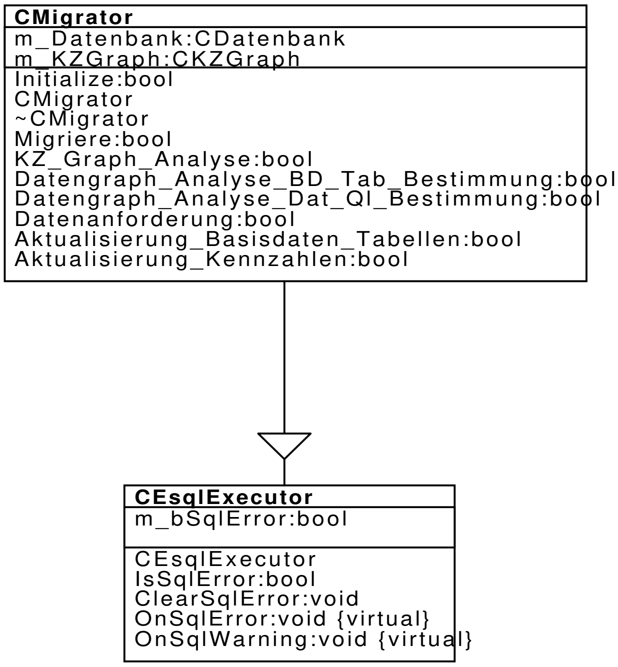


Abbildung 5-8 CMigrator-Klasse

Die Kernoperation von CMigrator ist “Migriere”. Sie bildet zusammen mit “Initialize” das *public interface* dieser Klasse; alle anderen Operationen sind *protected*. Ein Call-Diagramm dieser Operation ist in der Abbildung 5-9 dargestellt; es enstpricht weitgehend den Bubbles aus der Abbildung 3-8 DFD “DW_Aktualisierung_Ausgewaehlte_KZ”.

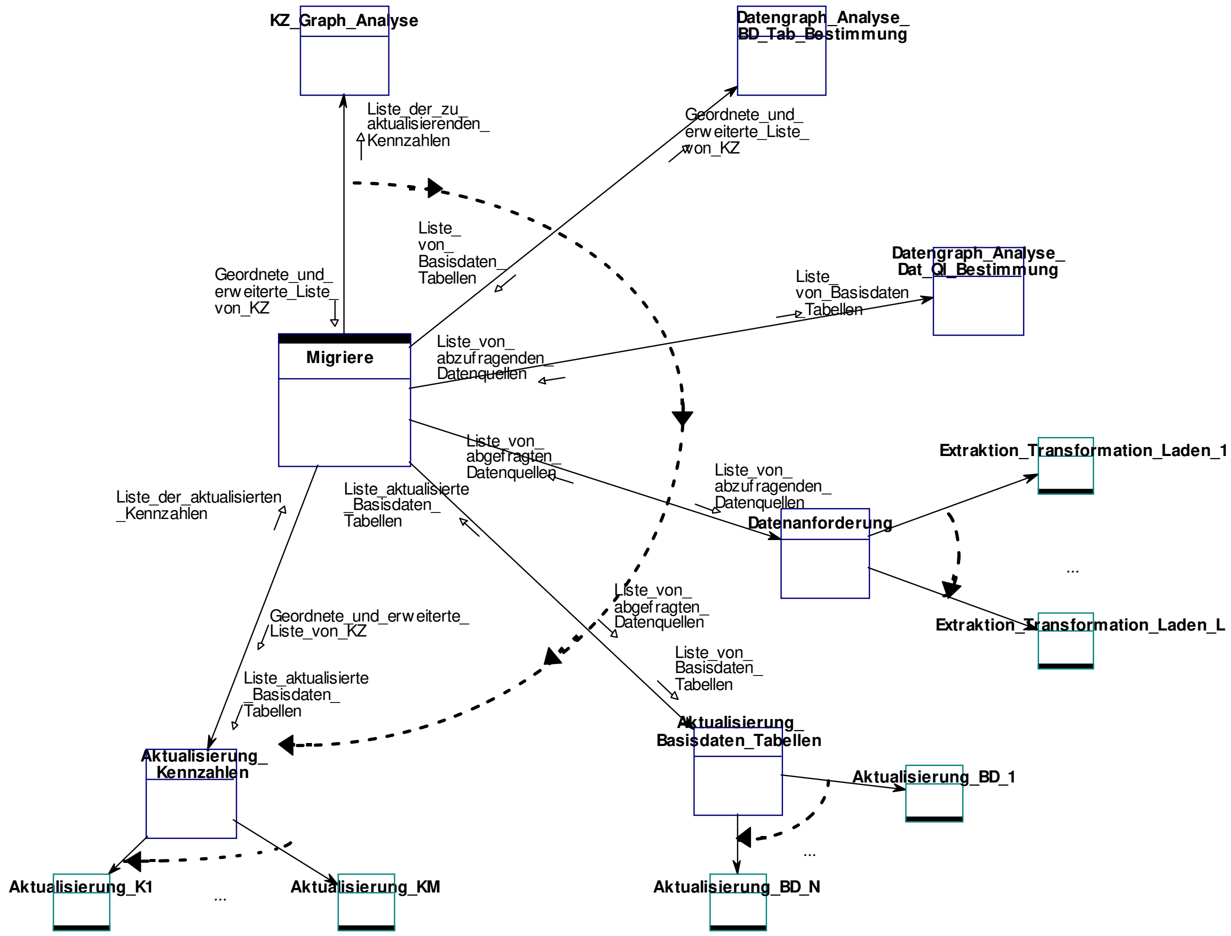


Abbildung 5-9 Call-Diagramm der Operation “CMigrator::Migriere”

Die Kästchen mit den Aufschriften “Extraktion_Transformation_Laden_…” , “Aktualisierung_BD_…” und “Aktualisierung_K…” sind nicht die Operationen von CMigrator, sondern es sind entweder externe Applikationen (wie im Falle der “Exraktion_Transformation_Laden…”-Rechtecke) oder Stored Procedures (“Aktualisierung_BD_…” und “Aktualisierung_K…”). Die Aufrufe dieser Programme bzw. Stored Procedures sind in dem Kennzahlengraphen und dem Datengraphen abgespeichert, die Klasse CMigrator muß nur eine einfache Funktionalität zum Anstoßen dieser Aufrufe implementieren. Ein Aufruf für ein externes Programm sieht folgendermaßen auf:

“/management/KZDW/datamigration/ArsTTApp -dwu…-dwp…-aru…-arp…-ars…”

Die Stored Procedures werden z.B. durch:

“BEGIN Aktualisiere_K1; END;”

aufgerufen.