In: Informationstechnik - Computer, Systeme, Anwendungen (Themenheft Wissensrepräsentation), 31. Jg., Heft 2, 1989, S. 141-154.

Klassische Datenmodelle und Wissensrepräsentation

Classical Data Models and Knowledge Representation

Theo Härder
Universität Kaiserslautern

Zusammenfassung

Künftig sollen Wissensbankverwaltungssysteme (WBVS) für alle Arten von wissensbasierten Anwendungen eine ähnliche Rolle spielen wie heute Datenbankverwaltungssysteme (DBVS) in betrieblichen Informationssystemen. In diesem Aufsatz wird am Beispiel des Relationenmodells untersucht, ob klassische Datenmodelle bereits den Modellierungsansprüchen von wichtigen Verfahren zur Wissensrepräsentation genügen oder ob sie zumindest als Basis für entsprechende Erweiterungen dienen können. Dazu skizzieren wir zunächst die Schlüsselkonzepte des Relationenmodells und diskutieren seinen Einsatz bei der Regelverarbeitung, zur Realisierung allgemeiner Abstraktionskonzepte sowie zur Behandlung von zeitlichen und räumlichen Zusammenhängen. Wegen der großen konzeptuellen Diskrepanz zwischen der Informationsmodellierung klassischer Datenmodelle und den semantisch weit höher angesiedelten Anforderungen typischer Verfahren der Wissensrepräsentation schlagen wir vor, daß in künftigen WBVS Datenmodelle mit "mehr Semantik und Objektorientierung" zur Unterstützung der Wissensrepräsentation eingesetzt werden.

Summary

In future, knowledge base management systems (KBMS) are supposed to play a similar role for all kinds of knowlege-based applications as database management systems (DBMS) in todays business information systems. Using the relational model as an example, this report investigates whether classical data models already satisfy the modeling needs of important knowledge representation methods or whether they may at least serve as starting point for appropriate extensions. For this purpose, we sketch the key concepts of the relational model and discuss its use for rule processing, for the realisation of general abstraction concepts as well as for handling of time-related and spatial relationships. A big conceptual gap is encountered between the poor semantic information modeling capabilities of classical data models and the rich modeling requirements of typical knowledge representation methods as far as semantic accuracy and completeness are concerned. Therefore we proposed that data models with "more semantics and object orientation" are used in future KBMS to support knowledge representation.

1. Was hat Wissensrepräsentation mit Datenbanken zu tun?

Datenintensive Rechneranwendungen vorwiegend in administrativ-betriebswirtschaftlichen Bereichen stellten schon immer hohe Anforderungen an ihre Datenhaltung. Etwa vor 25 Jahren führte die Beobachtung, daß die Aufgaben der Datenverwaltung in diesen Anwendungen sehr ähnlich waren, zu der Grundsatzforderung, alle Datenverwaltungsfunktionen aus den Anwendungsprogrammen herauszulösen, besonders effizient zu implementieren und in einem separaten Teilsystem bereitzustellen. Dadurch wurde die Entwicklung von DBVS angestoßen, durch die eine Verallgemeinerung und Standardisierung von Funktionen zur Definition und Manipulation sowie zur Überwachung vielfältiger Aspekte der Integrität und Zugriffskontrolle der Daten erreicht wurde. Im Laufe der Entwicklung setzten sich drei Typen von Anwendungsschnittstellen oder Datenmodellen eines DBVS durch, und zwar Netzwerk-, Hierarchie- und Relationenmodell [Da83]. Diese klassischen Datenmodelle existieren in einer Vielzahl von Implementierungen und haben sich - vorwiegend bei kommerziellen Anwendungen - in der Praxis bewährt.

In den letzten 10 Jahren ist ein verstärktes Forschungs- und Entwicklungsinteresse zu beobachten, eine Vielzahl neuer Anwendungsbereiche durch leistungsfähige Wissensbasierte Systeme (WBS) zu unterstützen, wobei der Einsatzschwerpunkt heute bei Expertensystemen liegt. Das Spektrum möglicher Anwendungen zeigt, daß vielfältige Strukturen zur Wissensrepräsentation bereitzustellen sind, um die Objekte und Wissensinhalte von diesen Anwendungsklassen durch ausdrucksstarke Konzepte und Mechanismen hinreichend vollständig und im erforderlichen Detail darstellen zu können. Heutige Systeme sind typischerweise zugeschnitten auf kleine Wissensbasen (die im Hauptspeicher verwaltet werden); ihre Realisierung ist oft stark durch die Eigenschaften einer Programmiersprache geprägt, und ihre Nutzung ist auf den Einbenutzerbetrieb beschränkt.

Mit der zunehmenden Komplexität wissensbasierter Anwendungen ist vorauszusehen, daß die Menge der an Entscheidungs- und Problemlösungsprozessen beteiligten Daten ohnehin sehr stark zunimmt. Deshalb wird man beim Entwurf von WBS nicht umhin kommen, auf bewährte DB-Techniken zurückzugreifen und sie in irgendeiner Form dem System verfügbar zu machen oder sie gar zu integrieren. Ein Kernkonzept, beide Technologien in einem System zu kombinieren, wird seit wenigen Jahren diskutiert: sogenannte Wissensbankverwaltungssysteme wurden in [Ke85] wie folgt definiert:

> **A system providing highly efficient management of large, shared knowledge bases for knowledge-directed systems.**

Die Grundidee ist hier die gleiche, die den Anstoß zur Entwicklung von DBVS gegeben hat. Durch das Herauslösen von (allgemeinen) Funktionen zur Wissensverwaltung und -arbeitung aus individuellen Anwendungen und durch die einheitliche Darstellung und Verwaltung der Wissensbasis bietet sich durch das WBVS-Konzept eine Weg an, wissensbasierten Anwendungen standardisierte Schnittstellen zu Systemen verfügbar zu machen, die Aufbau, Zugriff und Aktualisierung von großen, gemeinsam nutzbaren Wissensbanken in einfacher Weise gestatten. Insbesondere bei der Realisierung wichtiger Systemeigenschaften wie Mehrfachzugriff, Zugriffskontrolle, Integritätserhaltung oder Zugriff auf verteilte Wissensbasen müssen bewährte DBS-Konzepte herangezogen werden. Es ist zu erwarten, daß diese Vorgehensweise nicht nur das Problemlösungsvermögen und die Praxistauglichkeit wissensbasierter Anwendungen erheblich verbessert, sondern daß auch bei Informationssystemen mit diesem Ansatz zusätzliche Funktionalität und produktivere Nutzungsmöglichkeiten gewonnen werden.

Aus meiner Sicht ergibt sich der stärkste Zusammenhang zwischen "Wissensrepräsentation und Datenbanken" durch das WBVS-Konzept, durch das eine Integration aller Funktionen angestrebt wird, um beide Bereiche zu überdecken. Aus Platzgründen können in diesem Aufsatz nur Aspekte der Objekt- und Wissensmodellierung aus DB-Sicht untersucht werden - aber nicht gleichermaßen wichtige Fragen nach den Auswirkungen solcher Vorschläge auf Systemleistung, Verteilbarkeit, Mehrfachnutzung usw. Nach einer Grobcharakterisierung typischer Anforderungen an die Wissensrepräsentation soll zunächst geklärt werden, warum sich klassische Datenmodelle für den Einsatz in wissensbasierten Anwendungen als zu schwach erweisen und welche Erweiterungen zur effizienteren und effektiveren Unterstützung wünschenswert erscheinen. Die Frage, wie bestehende Überlappungen bei Modellen zur Daten- und Wissensrepräsentation beseitigt und in ein reibungsloses Zusammenspiel überführt werden können, charakterisiert eines der Kernprobleme hinsichtlich der Architektur von WBVS. Als Ausblick wird ein Lösungsweg in Form eines Architekturrahmens für solche WBVS skizziert.

2. Anforderungen an die Wissensrepräsentation

In [Wo83] wird betont, daß die Wahl einer geeigneten Repräsentation in wissensbasierten Systemen schwieriger als in anderen Rechneranwendungen ist, da das Spektrum der Möglichkeiten erheblich größer und die relevanten Kriterien weniger klar sind. Allgemeine Forderungen nach einem "guten WR-System mit einer geeigneten Menge von Darstellungsprimitiven zur Handhabung nicht-abgeschlossener Wissensbereiche" führen wegen ihres Mangels an Konkretkeit kaum zu kontroverser Diskussion. Konkrete Modellvorschläge zur (natürlichen, einheitlichen, verständlichen) Strukturierung eines WR-Systems, zur flexiblen Detailabstraktion, zur dynamischen Erweiterung und Anpassung oder zur effizienten Auswahl relevanter WR-Aspekte rufen dagegen viel leichter Widersprüche hervor, da sie direkt an einer Vielzahl wenig kohärenter Probleme und Situationen gemessen werden können. Es ist eben nicht einfach, in allgemeiner Weise festzulegen, was "expressive adequacy" und "notational efficacy" als wesentliche Aspekte für ein WR-System (nach [Wo83]) bedeuten. Davon zeugen auch die in diesem Themenheft (IT 1/89) gesammelten Beiträge. Sie beschreiben vielfältige Forschungsansätze, die als Suche nach den allgemeinen Prinzipien der WR verstanden werden können.

Wenn schon kein generell akzeptiertes WR-Modell existiert, so ist es doch für die Diskussion der Beziehungen zwischen "Datenmodell und Wissensrepräsentation" notwendig, Ansprüche und Anforderungen der WR in etwas konkreterer Weise festzulegen. Deshalb gehen wir von einem WR-Spektrum aus, das sich in seiner Ausdruckskraft durch folgende vier Techniken [My80] kennzeichnen läßt: Produktionsregeln, Logiksysteme erster Ordnung, Semantische Netzwerke und Frames. Diese Techniken - häufig auch als Mischformen eingesetzt - erlauben die Darstellung bzw. Nutzung von

- Objekten, Beziehungen und Aktivitäten
- allgemeinen Abstraktionskonzepten (Generalisierung, Aggregation, usw.)
- Ableitungsverfahren (explizit über Regeln und implizit über Vererbung [FK85]).

Weiterhin müssen Aufgaben wie Erwerb, Bereitstellung und Ableitung von Wissen abgedeckt werden. Darüber hinaus wird Meta-Wissen über Anwendungsbereich, Wichtigkeit, Genauigkeit und Zuverlässigkeit des gespeicherten Wissens ausgenutzt.

Allgemeingültige Modelle werden noch weitergehende Anforderungen an die Objekt- und Wissensrepräsentation erfüllen müssen. Aus unserer heutigen (von DBVS geprägten) Sicht betrifft das vor allem die

- Spezifikation und Handhabung komplexer Objekte sowie die Kontrolle komplexer Integritätsbedingungen
- Wartung und Auswertung zeitlicher und räumlicher Beziehungen.

Der selektive Einsatz verschiedener Techniken in heutigen WBS gibt einen deutlichen Hinweis darauf, daß keine dieser WR-Techniken ideal für alle Aufgaben ist. In Systemen, die komplexe Anwendungen unterstützen sollen, scheint es deshalb erforderlich zu sein, die Vorteile wie Einförmigkeit und eingeschränkte Komplexität, die sich aus dem Einsatz einer Methode ergeben, aufzugeben zugunsten multipler WR-Techniken, die auf unterschiedliche Teilaufgaben zugeschnitten sind. Deshalb ist zu erwarten, daß in WBVS mehrere sich ergänzende WR-Techniken unterstützt werden müssen, um Angemessenheit und Genauigkeit der Modellierung für die angestrebte Anwendungsbreite sicherzustellen.

3. Einsatz klassischer Datenmodelle

Klassische Datenmodelle erheben zwar den Anspruch, allgemeingültig zu sein - sie sind es aber bestenfalls für die Modellierung von administrativ-betriebswirtschaftlichen Anwendungen, die bei ihrem Entwurf Pate standen. Der Modellierungsbedarf solcher Anwendungsgebiete (wie er vor etwa 20 Jahren gesehen wurde) wirkte sich prägend auf die Datenmodellkonstrukte aus. Setzt man sie heute zur Objekt- und Wissensstrukturierung in WBS ein, so leisten sie bestenfalls für Teilaspekte eine wirksame Unterstützung. Die konzeptuelle Diskrepanz zwischen den Datenbankmodellkonstrukten und dem Modellierungsbedarf der Anwendung wird in der Regel schwerfällige, unvollständige und leistungsarme Anwendungsmodelle erzwingen.

Zur Konkretisierung dieser Aussagen beziehen wir uns beispielhaft auf das Relationenmodell. (Datenbanksysteme, die dieses Modell implementieren, setzen sich zunehmend auf dem Markt durch; die relationale Sprache SQL wurde international standardisiert.) Das Relationenmodell bildet Objekte und Beziehungen in einheitlicher Weise durch Relationen ab und stellt durchgängig das Konzept der Mengenorientierung bei Operationen zur Verfügung. Durch deskriptive (nicht-prozedurale) Sprachen für alle Aufgaben der Datenverwaltung wird eine Einflußnahme des Programmierers auf den Ablauf von Operationen (Art des Zugriffs, Reihenfolge und Vorgehensweise beim Suchen) verhindert. Durch seine Kerneigenschaften - Symmetrie der Darstellung von Objeken und Beziehungen sowie Verzicht auf "Navigation" durch den Programmierer - bleibt das Relationenmodell sehr einfach und behält (gerade deswegen) ein hohes Maß an Flexibilität und Datenunabhängigkeit.

3.1 Schlüsselkonzepte des Relationenmodells

Um die Unterschiede zu den Verfahren und Ansprüchen der Wissensrepräsentation herausarbeiten zu können, sollen zunächst die wichtigsten Konzepte des Relationenmodells zur Informationsmodellierung skizziert werden. Sie betreffen die Modellabbildung von Objekten (Entities) der realen Welt (Miniwelt), ihrer Beziehungen (Relationships) zueinander sowie der Erhaltung ihrer (im Modell festgelegten) Integritätsbedingungen beim Nachvollzug der Vorgänge der Miniwelt durch die Operationen des Datenmodells. Im Vergleich zu den Zielsetzungen der Wissensrepräsentation handelt es sich bei dieser Informationsmodellierung um eine eher semantikarme Darstellung von Weltausschnitten: das Relationenmodell erlaubt die Beschreibung von einfach strukturierten Objekten und von "syntaktischen" Beziehungen zwischen diesen Objekten; es übernimmt weiterhin die Gewährleistung einfacher Integritätsbedingungen.

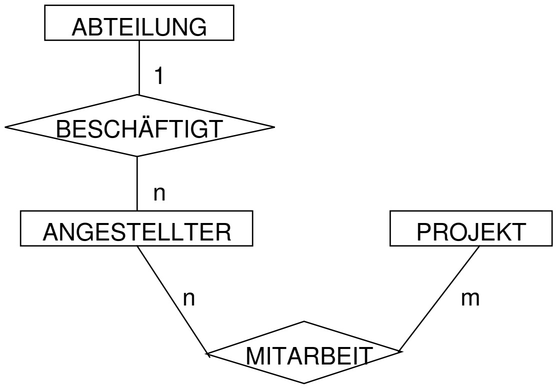

Bild 1: ER-Diagramm unserer Beispiel-Miniwelt

Informationsmodellierung

Die Art der Informationsdarstellung im Relationenmodell sei durch ein kleines Beispiel skizziert. Es seien in einer Miniwelt "Betrieb" Informationen über Abteilungen und ihre Angestellten sowie über Projekte, an denen die Angestellten mitarbeiten, zu speichern. Die bei der Informationsbedarfsanalyse erhobenen Objekt- und Beziehungstypen lassen sich in übersichtlicher Weise durch ein Entity-Relationship-Diagramm [Ch76] veranschaulichen. Unser Beispiel nach Bild 1 kann als eine typische DB-Modellierung (kleiner Ausschnitt) in Bezug auf Detaildarstellung und Genauigkeit der Information aufgefaßt werden. Aus diesem ER-Diagramm (in Bild 1 wurden der Übersichtlichkeit halber die Attribute weggelassen) kann unmittelbar die Relationendarstellung abgeleitet werden. Sie wird im DB-Schema für die konkrete DB-Installation festgeschrieben: die Beschreibung unseres Beispiels benötigt vier Relationen und hat vereinfacht folgendes Aussehen:

ABTEILUNG (<u>ABTNR</u>, BEZEICHNUNG, AORT)
ANGESTELLTER (<u>ANR</u>, NAME, EDATUM, BERUF, ABTNR)
MITARBEIT (<u>ANR, PNR</u>, ROLLE)
PROJEKT (<u>PNR</u>, BEZEICHNUNG, PROJEKTSUMME, AUFTRAGGEBER)

Die Attributnamen sind weitgehend selbsterklärend oder werden durch den Kontext klar. Durch die Unterstreichung von Attributen in Relationen werden hier die Primärschlüssel gekennzeichnet. ABTNR in ANGESTELLTER ist ein sogenannter Fremdschlüssel; mit seiner Hilfe und mit dem Primärschlüssel ABTNR in ABTEILUNG wird die (1:n)-Beziehung "Abteilungszugehörigkeit" nachgebildet. Die Projektmitarbeit von Angestellten ist in Bild 1 als eine (n:m)-Beziehung definiert. Aus diesem Grund wird sie als separate Relation MITARBEIT realisiert. Über die Primär-Fremdschlüsselbeziehung zwischen ANGESTELLTER und MITARBEIT sowie PROJEKT und MITARBEIT wird die ursprüngliche (n:m)-Beziehung aufgelöst in zwei funktionale Beziehungen. Als strukturelle Integritätsbedingungen garantiert das Relationenmodell die Einhaltung der beiden relationalen Invarianten: Primärschlüsselbedingung, die die Eindeutigkeit der Tupeln in einer Relation gewährleistet, und Fremdschlüsselbedingung, die überwacht, daß stets zu jedem Wert von ABTNR in ANGESTELLTER ein gleicher Wert für ABTNR in ABTEILUNG vorliegt. Entsprechendes gilt für ANR in MITARBEIT und ANGESTELLTER sowie PNR in PROJEKT und MITARBEIT.

Relationale DB-Sprachen sind sehr ausdrucksstark und verkörpern ein Auswahlvermögen, das äquivalent dem des Prädikatenkalküls erster Ordnung ist. Darüber hinaus besitzen sie oft eine Reihe zusätzlicher Konzepte, die sich an den Erfordernissen praktischer Anwendungen orientieren: Sortierung, Gruppenbildung, arithmetische Ausdrücke, Built-in-Funktionen, Maskenbildung, Behandlung von Nullwerten usw. [Ch80].

Eine einfache Anfrage, durch die Programmierer (mit einigen Attributen) mit dem Abteilungssitz München ausgewählt werden sollen, hat in SQL folgendes Aussehen:

```
SELECT    ANR, NAME, BERUF AORT
FROM      ABTEILUNG A, ANGESTELLTER B
WHERE     A.ABTNR = B. ABTNR           AND
          BERUF = 'Programmierer'      AND
          AORT = 'München'
```

Bei dieser Formulierung bezieht man sich auf die im DB-Schema vereinbarten Relationen- und Attributnamen (und ggf. Namen von temporären Objekten) - den sogenannten Beschreibungs- oder Metadaten. Typübergreifende Operationen (Verbund) werden ausschließlich durch Wertegleichheit in den Verbundattributen der beteiligten Relationen (A.ABTNR = B.ABTNR) spezifiziert.

<u>Sichtkonzept</u>

Das Ergebnis jeder Relationenoperation ist wiederum eine Relation (Hüllenoperation). Solche Relationen können zum eigenständigen Objekt - einer sogenannten Sicht - erklärt werden und wiederum Ziel von Anfrage- und (in eingeschränker Weise) Manipulationsoperationen sein. Im Gegensatz zu den extensionalen Relationen (abgespeicherte Relationen) lassen sich solche Sichten als intensionale Relationen (abgeleitete Relationen) auffassen.

Zur Sichtdefinition kann die volle Auswahlmächtigkeit der Anfragesprache ausgenutzt werden. Das bedeutet, daß sich Sichten über beliebig viele Relationen und Sichten erstrecken und aggregierte Daten beinhalten können. Beispielsweise könnte man die obige Frage zur Definition einer Sicht 'MüNCHNER_PROGRAMMIERER' (genauer: die in Münchner Abteilungen beschäftigten Programmierer) heranziehen.

Das Sichtkonzept hat sich für die relationale DB-Verwaltung als sehr leistungsstark erwiesen. Es unterstützt die flexible Definition von angepaßten Anwendungsschemata (Externe Sichten) und fördert wesentlich die Datenunabhängigkeit - ein gewisses Maß an logischer Datenstrukturunabhängigkeit läßt sich sogar erreichen: eine Umstrukturierung der Relationen kann den Anwendungen verborgen werden, wenn die ursprünglichen Relationenreferenzen als Sichtreferenzen erhalten bleiben [Ch80].

<u>Integritätszusagen</u>

Neben den bereits erwähnten relationalen Invarianten, die anwendungsunabhängig die Einhaltung struktureller Modellbeziehungen gewährleisten, sieht das Relationenmodell die Spezifikation einfacher anwendungsbezogener Integritätsbedingungen (manchmal auch semantische Integritätsbedingungen genannt) vor. Sie betreffen typischerweise zulässige Werte oder Wertkombinationen in Tupeln oder inhaltliche Beziehungen zwischen Tupeln oder in einer Tupelmenge. Als Beispiel in SQL-Notation diene folgende leicht verständliche Formulierung:

ASSERT A1 ON PROJEKT: PROJEKTSUMME ≥ '100K' AND PROJEKTSUMME ≤ '10M'

Integritätsbedingungen erlauben die Kontrolle von Abhängigkeiten, Einschränkungen und Gesetzmäßigkeiten zwischen den Objekten und Beziehungen der Miniwelt in der Datenbank. Bei jeder Änderung der Daten kann so überprüft werden, ob die im DB-Schema spezifizierten Bedingungen erfüllt bleiben; ansonsten muß die Operation (ggf. verzögert im Rahmen der Transaktion) zurückgewiesen werden. Oft werden Integritätsbedingungen dazu benutzt, um redundante Daten und Beziehungen in der DB zu kontrollieren. In solchen Fällen kann eine Integritätsbedingung auch als Ableitungsregel aufgefaßt werden, die aktiv zur Integritätserhaltung genutzt werden kann. Durch das sogenannte Triggerkonzept werden durch Regeln spezifizierte Folgeänderungen ausgelöst.

Wir können also festhalten, daß das Relationenmodell bereits zwei Konzepte zum Umgang mit ableitbarer Informationen anbietet:

- Intensionale Relationen lassen sich mit Hilfe von Sichten definieren.
- Die Einhaltung von Integritätsbedingungen wird bei jeder DB-Aktualisierung erzwungen; durch das Triggerkonzept lassen sich redundante Daten über eine Art Regelverarbeitung ("forward reasoning") nachführen.

3.2 Prämissen der Anwendbarkeit

Es wurde bereits betont, daß die klassischen Datenmodelle im Hinblick auf ihre Nutzung in bestimmten Anwendungsklassen entworfen worden sind. Dabei wurde von einer Reihe inhärenter Anwendungsannahmen ausgegangen, die ihre Tauglichkeit und Flexibilität bei Aufgaben der Wissensrepräsentation wohl erheblich einschränken. Diese Annahmen sind fest in den Datenmodellen verwurzelt, so daß sie weder eliminiert noch beim konkreten Einsatz umgangen werden können. Obwohl bisher eine Reihe von Ansätzen zur Nutzung klassischer Datenmodelle für verschiedene Aspekte der Wissensrepräsentation [JV83] untersucht und empfohlen wurden, muß bezweifelt werden, daß diese Modelle einen allgemeingültigen Ausgangspunkt für einen breiten Einsatz bei der Wissensrepräsentation verkörpern. Das gilt sowohl wegen eines Mangels an geeigneten Konzepten als auch wegen der vielen inhärenten Anwendungsprämissen. Zur Verdeutlichung seien hier die wichtigsten Prämissen für eine angemessene Informationsmodellierung aufgezählt [Hä88, Re87]:

1. Es ist ein hoher Grad an Wissen über die Struktur (zukünftig) darzustellender Sachverhalte/ Objekte erforderlich, um das DB-Schema entwerfen zu können. Schemaergänzungen in beschränkem Umfang sind zwar möglich, jedoch zumindest mühsam; Strukturänderungen (wenn überhaupt möglich) erzeugen meist prohibitiven Aufwand.
2. Für die Daten eines Objekttyps ist eine feste, vordefinierte Struktur vorgesehen, die im DB-Schema vereinbart ist.
3. Jedes Tupel (jede Ausprägung eines Objekttyps) repräsentiert in der Regel genau ein bestimmtes Objekt der Anwendung (ABTEILUNG, PROJEKT usw.); seine Attribute sind im Prinzip einwertig und enthalten genau das und nur das zum jeweiligen Zeitpunkt gültige Wissen über das Objekt.
4. Es ist eine einheitliche, eindeutige und dauerhafte Bezeichungsweise für ein Objekt zu wählen (Primärschlüssel). Synonyme werden in der Regel nicht akzeptiert.
5. Die Zahl der darzustellenden Objekttypen ist relativ gering ($\leq 10^2$, nicht 10^4), die Zahl der erwarteten Ausprägungen eines Typs im Vergleich dazu relativ groß (nicht 10^2, sondern $\leq 10^6$).
6. Der DB-Schemaentwurf führt auf eine große Homogenität (oder gar Gleichheit) innerhalb eines Objekttyps und in der Regel auf eine große Inhomogenität zwischen Objekttypen.
7. Objekttypen sind semantisch disjunkt, was notfalls durch den Schemaentwurf zu erzwingen ist. Generalisierungshierarchien werden nicht unterstützt; Mehrfachzuordnung eines Objektes zu verschiedenen Typen ist verboten.

Es fällt auf, daß von relativ starren und einfachen Informationsstrukturen ausgegangen wird und daß vor allem die typischen Mengengerüste kaum mit denen von WR-Welten übereinstimmen dürften. Weiterhin wird sehr strikt zwischen Objekttypen und -ausprägungen (Metainformation vs. Benutzerinformation) unterschieden, was einer Einführung von Abstraktionskonzepten in diesen Datenmodellen im Wege steht.

Schränken diese modellgebundenen Anwendbarkeitsannahmen einen möglichen Einsatz der klassischen Datenmodelle für Aufgaben der Wissensrepräsentation schon erheblich ein, so darf nicht vergessen werden, daß reale Systeme zusätzliche Schwächen besitzen. Deshalb ist diese Liste von Einsatzprämissen zu komplementieren durch Einschränkungen, die konkrete DBVS-Implementierungen aufweisen (Schema-Anpassungen, maximale Länge eines Attributes, eines Tupels, einer Datei usw.). Dieses facettenreiche Thema soll hier nicht vertieft werden; es sei lediglich ganz allgemein festgestellt, daß so wichtige Systemeigenschaften wie Flexibilität und Erweiterbarkeit in praktischen Implementierungen oft ganz kleingeschrieben werden.

4. Wissenrepräsentation mit dem Relationenmodell?

Es ist offensichtlich, daß mit den bisher diskutierten Konzepten und Beschränkungen klassischer Datenmodelle nur wenige Anforderungen von WR-Systemen - vor allem was Angemessenheit und Genauigkeit der Modellierung betrifft - erfüllt werden können. Deshalb soll zunächst untersucht werden, ob und wie sich Darstellungsvermögen und Ausdruckkraft des Relationenmodells durch Erweiterungen und Ergänzungen verbessern läßt und wo ggf. seine inhärenten Grenzen liegen.

4.1 Regelverarbeitung unter Nutzung des Sichtkonzeptes

Als ersten Aspekt wollen wir die Möglichkeiten der Regelverarbeitung im Relationenmodell diskutieren. Zur Formulierung des Regelsystems verwenden wir die bekannte Notation definiter Horn-Klauseln. Wie üblich, bezeichnet P(x,y,z) ein (in diesem Fall dreistelliges) Prädikat, mit dem Prädikatsnamen P und den Variablen x,y,z, die frei oder bereits an bestimmte Werte gebunden sein können. Regeln lassen sich in allgemeiner Form durch Horn-Klauseln ausdrücken:

$$R \leftarrow P_1 \wedge P_2 \wedge \ldots \wedge P_k$$

Das Erfülltsein aller Prädikate P_1 bis P_k impliziert das Erfülltsein des Prädikates R, d.h., aus der wahren Vorbedingung kann die Schlußfolgerung abgeleitet werden. Aussagen, die ohne Vorbedingungen immer erfüllt sind, lassen sich mit leerer rechter Seite als Horn-Klauseln darstellen. Folglich erhalten wir für die Fakten einer Anwendung folgende Notation:

$$R(x,y,z) \quad \leftarrow$$

Zur Diskussion der Regelverarbeitung beziehen wir uns auf das Miniweltbeispiel in Bild 1.

Die fest vorgegebenen Informationen - die vier Relationen des DB-Schemas - lassen sich direkt als Hornklauseln ausdrücken, also z.B.

$$\text{ANGESTELLTER}(s, t, u, v, w) \quad \leftarrow ,$$

wobei allerdings die Position der Variablen von Wichtigkeit ist, da die Attributnamen bei dieser Darstellung nicht übernommen werden. Ableitbare Informationen werden mit Hilfe von allgemeingültigen Regeln, die über der Faktenmenge auszuwerten sind, spezifiziert.

Die Eigenschaften, ein erfahrener Projektleiter (ERFPROJEKTLT) zu sein, sei durch folgende Bedingungen festgelegt:

- mindestens fünfjährige Firmenzugehörigkeit und
- Leiter eines wichtigen Projektes, wobei hier "wichtig" definiert ist durch "spezieller Auftrag" oder "Projektsumme > 1 Mio.".

Als Regeln formuliert (mit Blick auf unser DB-Schema) ergibt sich:

1.	WICHTIGES_PROJEKT (p)	$\leftarrow$	PROJEKT (p, -, -, q) Λ SPEZAUFTRAG(q)
2.	WICHTIGES_PROJEKT(p)	$\leftarrow$	PROJEKT (p, -, q, -) Λ (q > '1M')
3.	BERUFSERFAHREN(s, t)	$\leftarrow$	ANGESTELLTER(s, t, u, -, -) Λ (JDIFF(HEUTE, u) > 5)
4.	ERFPROJEKTLT(s, t)	$\leftarrow$	BERUFSERFAHREN(s, t) Λ MITARBEIT (s, u, 'Leiter') Λ WICHTIGES_PROJEKT(u)

Der Term JDIFF(HEUTE,u) ist sicher bei funktionsfreien Horn-Klauseln in dieser Form nicht zulässig. Der Einfachheit halber nehmen wir jedoch an, daß wir eine Standardfunktion HEUTE zur Bestimmung des Tagesdatums und einer Funktion JDIFF, die die Jahresdifferenz berechnet, zur Verfügung haben.

Es wurde schon darauf hingewiesen, daß mit Hilfe des Sichtkonzepts ableitbare Information spezifiziert werden kann. Über dieses Sichtkonzept läßt sich nun der Zusammenhang zwischen Relationenmodell und Regelverarbeitung leicht herstellen. In Bild 2 ist die Regel zur Ableitung erfahrener Projektleiter mit Hilfe von Sichten in SQL formuliert. Die Sichten dienen hier nur der übersichtlicheren Darstellung; die Sichtdefinitionen hätten natürlich in Bild 2c direkt verwendet werden können. Andererseits kann die Anfrage 2c wiederum zur Sicht erklärt (DEFINE VIEW ERFPROJEKTLT AS) und direkt zu weiteren Anfragen oder Regelauswertungen genutzt werden.

Da relationale Sprachen keine rekursiven Anfragen zu formulieren gestatten, ist die Auswertung rekursiver Regeln (zunächst) nicht möglich. Es gibt jedoch zahlreiche Ansätze zur Spracherweiterung und zur Integration der Rekursion [Ba85], so daß dieser Aspekt im Relationenmodell relativ einfach berücksichtigt werden kann.

```
DEFINE VIEW WICHTIGES_PROJEKT AS

SELECT   PNR
FROM     PROJEKT
WHERE    AUFTRAGGEBER IN
         (SELECT AUFTRAGGEBER
         FROM SPEZAUFTRAG)
UNION
SELECT   PNR
FROM     PROJEKT
WHERE    PROJEKTSUMME > '1 M';
```

a) Formulierung der "Wichtigen Projekte" als Sicht (Regeln 1 und 2) (SPEZAUFTRAG sei als einstellige Relation vorhanden)

```
DEFINE VIEW BERUFSERFAHREN AS

SELECT   ANR, NAME
FROM     ANGESTELLTER
WHERE    JDIFF (HEUTE, EDATUM) > 5;
```

b) Sicht auf "Berufserfahrene Angestellte" (Regel 3)

```
SELECT ANR, NAME
FROM      BERUFSERFAHREN X, MITARBEIT Y, WICHTIGES_PROJEKT Z
WHERE     X. ANR    = Y. ANR     AND
          Y. PNR    = Z. PNR     AND
          Y. ROLLE  = 'Leiter';
```

c) Formulierung der Sicht "Erfahrene Projektleiter" (Regel 4) mit Hilfe der ersten beiden Sichten

Die Planung solcher Ableitungen läßt sich als "backward reasoning" auffassen; die Materialisierung einer Sicht (Ausführung der Relationenoperationen) dagegen entspricht dem "forward reasoning".

Bild 2: Beispiel für regelbasierte Anfragen in SQL

Ohne weitere Diskussion sei hier nur festgestellt, daß Regelverarbeitung durch das Relationenmodell unterstützt wird. Natürlich würde sein Einsatz bei regelbasierten Anwendungen dynamische Erzeugung und effiziente Handhabung großer Mengen von Sichten voraussetzen - ein Aspekt, der wohl bisher bei keiner Implementierung eines relationalen DBS im Vordergrund stand.

4.2 Fehlen allgemeiner Abstraktionskonzepte

Die Verfügbarkeit von leistungsfähigen Abstraktionskonzepten für die Darstellung von Objekten und ihren Beziehungen ist eine der Hauptforderungen an ein WR-System. Vier voneinander unabhängige Abstraktionskonzepte werden in der Literatur [My80,] als wünschenswerte und hilfreiche Abbildungskonzepte herausgestellt. Einige bekannte WR-Modelle - vor allem semantische Netze und Frame-Ansätze - enthalten sie bereits - allerdings mit unterschiedlicher Gewichtung und Vollständigkeit.

Klassifikation und Generalisierung

Durch das Konzept der Klassifikation werden Objekte mit gleichen oder ähnlichen Eigenschaften zu einem Objekttyp, der die gemeinsamen Eigenschaften aller Objektausprägungen verkörpert, zusammengefaßt. Die Beziehung zwischen Ausprägung und Typ ("instance-of") entspricht im Relationenmodell der von Tupeln und Relationen. Da keine flexible Nutzung der Relationenbeschreibung (Metadaten) durch die DB-Sprache (z.B. SQL) vorgesehen ist, wird das Konzept der Klassifikation bestenfalls teilweise unterstützt.

In Bild 1 wurde ein Objekttyp ANGESTELLTER eingeführt, wobei unterstellt wurde, daß alle Angestellte in gleichartiger Weise beschrieben werden. Möchte man nun Sekretärinnen, Ingenieure, Facharbeiter usw. zugeschnitten auf ihre spezielle Situation beschreiben, so treten neben einigen gemeinsamen Attributen (ANR, NAME usw.) eine Reihe von speziellen Attributen auf (z.B. FACHRICHTUNG und DIPLOM bei Ingenieuren). Das Relationenmodell bietet nun für eine solche Informationsmodellierung keine geeigneten Konzepte an. Es erzwingt im allgemeinen die Einführung von separaten Relationen für Sekretärinnen, Ingenieure usw., wobei jedoch die Tatsache, daß es sich bei allen von Angestellten handelt, verlorengeht, wenn nicht zusätzliche Redundanz eingeführt wird. Von Natürlichkeit der Modellierung kann keine Rede mehr sein [SS77].

Durch das Konzept der Generalisierung kann eine spezielle Beziehung ("is-a") zwischen den "spezialisierten" und dem zugehörigen "generischen" Objekttypen definiert werden (Bild 3). Durch wiederholte Anwendung dieses Konzeptes lassen sich beliebige Generalisierungshierarchien bilden (Angestellte und Arbeiter sind Firmenangehörige). Information, die die gesamte Klasse betrifft, läßt sich an den generischen Objekttyp binden und zur Beschreibung der Objekte in der Hierarchie nutzen. Seine Ausdrucksmächtigkeit gewinnt das Konzept durch die Möglichkeit der Attributvererbung über die "is-a"- und "instance-of"-Beziehungen [Br83].

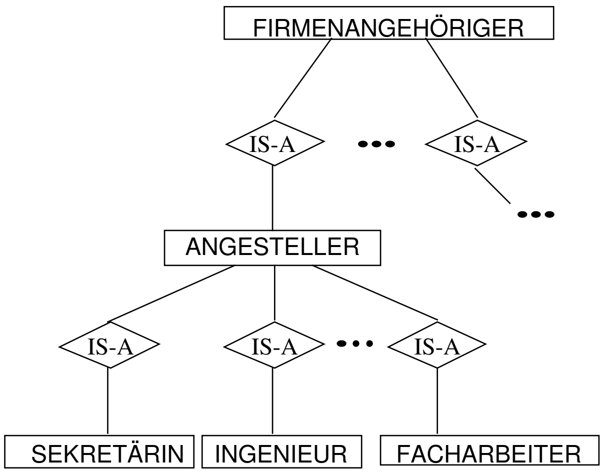

Bild 3: Generalisierungshierarchie als ER-Diagramm

Das Generalisierungskonzept, das einen hohen Stellenwert für die WR besitzt, läßt sich in einfacher Weise nicht in das Relationenmodell integrieren. Klassische Datenmodelle verlangen generell eine Vorabdefinition der Objekttypen (Relationen) im DB-Schema. Sie legen die zugehörige Beschreibungsinformation als Meta-Information in speziellen Katalogen ab und behandeln sie "anders" als Objektausprägungen (Tupeln). In der Regel bietet die DB-Sprache dem Benutzer keine Möglichkeit, direkt auf Meta-Informationen zuzugreifen, d.h., Fragen, die Typinformation betreffen, zu stellen. Zur Erhöhung der Flexibilität der DB-Interaktion haben bestimmte Systeme spezielle Sprachklauseln als Zusatzmechanismen hinzugefügt, um beispielsweise Zugriff auf DB-Schema-Information zur Laufzeit zu ermöglichen.

Die Trennung von Typ und Ausprägung ist fest in der Grundphilosophie des Relationenmodells verankert. Obwohl Meta-Informationen oft auch als "Relationen" (gleiche Speicherungsstrukturen) abgelegt werden, bleibt ihre Anbindung an die DB-Sprache ausgeschlossen. Diese Unsymmetrie bei Typ- und Ausprägungsinformation beschränkt die Ausdrucksmächtigkeit solcher Datenmodelle und verhindert ihre flexible Erweiterung sowie dynamische Restrukturierung. Die Eliminierung der Trennung von Typ und Ausprägung - Typinformation kann als Ausprägungsinformation einer abstrakteren Ebene gesehen werden - ist wesentliche Voraussetzung, bestimmte WR-Strukturen angemessen unterstützen zu können.

Aggregation

Mit Hilfe des Aggregationskonzeptes lassen sich (Komponenten-) Objekte zu einem neuen Objekt, dem Aggregat, zusammensetzen. Die innere Struktur wird explizit mit Hilfe der "component-of"-Beziehung (auch "part-of") bekanntgemacht, so daß bei solchen Objekten eine Detailsicht als auch eine Ganzheitssicht eingenommen werden kann. Durch Wiederholung von solchen Aggregationsschritten läßt sich diese "strukturelle Objektorientierung" oder diese "Bildung von komplexen Objekten" auf Aggregationshierarchien von Objekten ausdehnen.

Der Mangel an Objektorientierung im Relationenmodell und seine Konsequenzen für die Handhabung komplexer Anwendungsobjekte lassen sich recht drastisch bei der Modellierung von 3D-Körpern veranschaulichen. Auf der Anwendungsebene möchte man beispielsweise eine Bohrung an einem Werkstück anbringen; nach dem Volumenmodell (CSG [RV82]) könnte dieser Vorgang durch "Subtraktion" eines Zylinders vom bisher konstruierten Körper beschrieben werden. Da das Relationenmodell nur Operationen auf den in DB-Schema erklärten Relationen kennt, müssen folglich solche höheren Operationen in äußerst umständlicher Weise nachgebildet werden.

Zur Illustration nehmen wir an, daß eine Begrenzungsflächendarstellung (BREP) vorliegt, für die in Bild 4 eine ER-Darstellung gewählt wurde. Je nach Reichhaltigkeit der Modellierung ist das DB-Schema eines 3D-Körpers nach dem Begrenzungsflächenmodell aus 10 oder mehr Relationen aufzubauen [Hä89]. Das bedeutet, daß die Einzelteile des Körpers - Punkte, Kanten, Flächen und Teilkörper - sowie ihre Beziehungen zueinander durch "unabhängige" Tupeln verschiedenen Typs zu beschreiben sind. Auf der Ebene des Datenmodells wird folglich ein komplexes Objekt durch eine heterogene Tupelmenge verkörpert, die lediglich in vielfältiger Weise über Wertegleichheit von Attributen (Primärschlüssel-Fremdschlüssel) verknüpft ist. Als Konsequenz dieser "Modellabbildung" verbleibt dem das Datenmodell realisierende DBVS nur die "atomisierte" Sicht einzelner Tupeln; die ganzheitliche Sicht des 3D-Körpers und damit die Möglichkeit seiner integrierten Behandlung sind verlorengegangen. Objektzugriffe (durch Folgen komplexer Verbundoperationen), Kontrolle von Integritätszusicherungen, Objektmanipulationen gemäß dem (semantisch weiter höheren) Anwendungsmodell müssen in recht aufwendiger Weise mit den verfügbaren Operationen des Relationenmodells - oft durch Tausende von Operationen - von der Anwendung nachgebildet werden

Es ist festzuhalten, daß das Relationenmodell kein - auch nicht ansatzweise - wie auch immer geartetes Aggregationskonzept besitzt. Ein solches Konzept, das auch durch spezielle Vererbungsmechanismen [Ma88a] angereichert werden kann, erscheint unabdingbar für die Objekt- und Wissensrepräsentation, da es unmittelbar auf eine strukturelle Objektorientierung [Di86] abzielt.

Assoziation

Die Assoziation erlaubt über eine definierte Beziehung "element-of" eine Menge von Objekten potentiell verschiedenen Types zu einem semantisch höheren Objekt zu vereinigen [Ma88a]. Das Konzept der Assoziation ist nicht auf die Gruppenbildung von Objekten mit ähnlichen Eigenschaften (Klassifikation) ausgerichtet, sondern vielmehr darauf, Gruppen mit heterogenen Objekten für einen bestimmten Kontext zusammenfassen zu können. Wiederholte Anwendung dieses Konzeptes erzeugt eine Assoziationshierarchie. Auch für dieses Konzept bietet das Relationenmodell nicht einmal Ansatzpunkte zu seiner Unterstützung oder Integration.

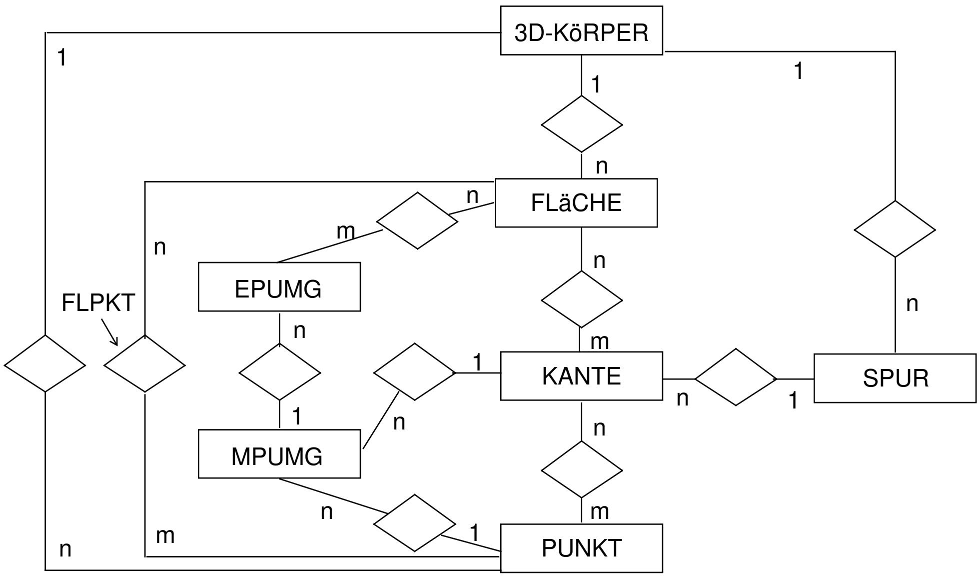

Bild 4: ER-Diagramm für eine Begrenzungsflächendarstellung von Werkstücken
(EPUMG und MPUMG modellieren spezielle Punktumgebungen)

4.3 Modellierung von Zeit und Raum

Neben den vier allgemeinen Abstraktionskonzepten spielen die Modellierung (und Überwachung) zeitlicher und räumlicher Zusammenhänge eine wichtige Rolle bei der Wissensrepräsentation. Es läßt sich generell feststellen, daß klassische Datenmodelle nicht einmal rudimentäre Mechanismen dafür aufweisen. Für das Relationenmodell werden jedoch in letzter Zeit einige darauf abzielende Erweiterungen diskutiert.

Modellierung temporaler Aspekte

Bei Änderungen von Objekten im Relationenmodell wird der alte Wert in Tupeln jeweils überschrieben. Ihre zeitbehaftete Darstellung (als Objektgeschichte) läßt sich lediglich über benutzerdefinierte Attribute, deren Wert die Anwendung als Zeitstempel interpretiert, simulieren. Weder bei der Darstellung noch bei der Verwaltung und Integritätserhaltung zeitlicher Aspekte bietet das Relationenmodell eine Hilfe an. Beispielsweise wird nach dem DB-Schema von Bild 1 in ANGESTELLTER immer nur der momentane Wert für Beruf und Abteilungszugehörigkeit vermerkt.

Ein temporales Relationenmodell würde voraussetzen, daß jede Änderung eines Objektes als jeweils neuer Zustand zusammen mit allen seinen bisherigen Zuständen als Objektgeschichte verwaltet wird. Diese Forderung impliziert, daß es (im Prinzip) keine Lösch- und Änderungsoperationen gibt, sondern lediglich eine Ergänzungsoperation. Einzelheiten der Geschichtsdarstellung, korrektur oder -ergänzung sowie die Problematik der Zeitaufzeichnung (Abweichungen zwischen Gültigkeits-, Bestimmungs- und Aufzeichnungszeit) können hier nicht betrachtet werden [Kl83]. Um die Ausdrucksmächtigkeit zeitlicher Anfragen zu illustrieren, skizzieren wir entsprechende Sprachklauseln als Ergänzung für SQL [Kä88, Sn87]. Es werden vier Anfragearten vorgeschlagen, die durch spezielle Prädikate (AT, WHEN, DURING, WHILE), die den Zeitbezug herstellen, formuliert werden. Prädikate, die sich explizit auf einen Zeitpunkt bzw. ein Zeitintervall beziehen, sind

- (B) AT <time>
- (B) SOMETIMES/ALWAYS DURING <interval>

wobei B eine boolesche Bedingung ohne Zeitbezug ist.

Eine implizite Festlegung der Zeit kann über

• WHEN (B)
• WHILE (B)

erreicht werden, wobei alle Zeitpunkte bzw. Zeitintervalle, in denen B erfüllt ist, ausgewertet werden. Ein einfaches Beispiel ist die Frage nach Abteilungsnummer und -ort vom Angestellten Meier am 1.9.1985:

```
SELECT   NAME, ABTNR, AORT
FROM     ANGESTELLTER X,ABTEILUNG Y
WHERE    (X. ABTNR      = Y. ABTNR
         AND X. NAME    = 'Meier')   AT  01/09/1985
```

Darstellung und Verwaltung räumlicher Objekte

Wie schon erwähnt, dürfen wir uns vom Relationenmodell keine Hilfe bei räumlichen oder geometrischen Aufgaben erhoffen. Die Diskussion soll hier lediglich das Problem bewußt machen und ggf. auf einen Lösungsweg hindeuten.

Die Aggregationsbeziehung hilft bei räumlichen Fragestellungen in allgemeinen nicht, da nur strukturelle Eigenschaften ausgenutzt werden können. Durch sogenannte "implied predicates" [Ro87] können jedoch in speziellen Fällen geometrische Zusammenhänge über vorgegebene "part-of"-Beziehungen abgeleitet werden (z.B. muß die Fläche eines Chips größer oder gleich der Flächensumme seiner Subzellen sein, die den Chip aufbauen).

Um die Komplexität der Fragestellung genauer ermessen zu können, versuchen wir einmal im Relationenmodell folgendes räumliche Problem zu lösen: Im ER-Diagramm nach Bild 3 sind ebene Flächen in einem 3D-Raum mit Hilfe mehrerer Relationen FLÄCHE, KANTE, PUNKT usw. beschrieben. Es sind alle Flächen gesucht, die in einem vorgegebenen quaderförmigen Raumausschnitt Q ganz enthalten sind.

Folgende Schritte sind zur Lösung erforderlich:

1. Da das Relationenmodell keine Objektorientierung aufweist, muß eine Fläche mit ihren Eckpunkten explizit abgeleitet werden. Von vorneherein können keine Ausschlußbedingungen für bestimmte Flächen angewendet werden, da die geometrische Information in den Koordinatenwerten in PUNKT steckt. Es sind also Verbundoperationen über FLÄCHE - FLPKT - PUNKT für jedes Flächentupel erforderlich.

2. Mit Hilfe der abgeleiteten Eckpunkte (X-, Y-, Z-Koordinate) einer Fläche ist ihr Enthaltensein in Q zu überprüfen. Hier wird die Schwierigkeit der SQL-Formulierung besonders deutlich, da in der WHERE-Klausel nur der Vergleich einzelner Koordinatenbeziehungen möglich ist. Es ist nicht klar, zu welcher Komplexität hierbei die Vergleichsbedingungen, die ohne zusätzliche arithmetische Funktionen und Beachtung geometrischer Zusammenhänge gar nicht formulierbar sind, ausarten. Die WHERE-Klausel "explodiert" vor lauter Koordinatenvergleichen und ist nicht mehr überschaubar. (In [Re87] wir ein (wesentlich einfacheres) zweidimensionales Beispiel diskutiert, bei dem die WHERE-Klausel-Bedingung 8 Zeilen umfaßt und durch Einsatz einer Berechnungsfunktion schon erheblich vereinfacht wurde).

Es ist offensichtlich, daß die vom Relationenmodell erzwungene Vorgehensweise kein Weg zur Lösung räumlicher Probleme darstellt. Die fehlende Objektorientierung erfordert zunächst umständliche Ableitungsschritte der räumlichen Objekte. Die Formulierung räumlicher Vergleiche ist aus der Sicht des Benutzers unnatürlich und kompliziert und somit in höchstem Maße fehleranfällig.

Eine Systemunterstützung des Benutzers ist bei solchen räumlichen Aufgabenstellungen erst möglich, wenn Objektzusammenhänge im DB-Schema spezifiziert werden können. Objekttyp-spezifische Operationen lassen sich dann zur Verarbeitung und Integritätsüberwachung der verschiedenartigen Objekte heranziehen. Auf diese Weise ist es dann auch möglich, die Behandlung räumlicher Zusammenhänge zu integrieren. Die Definition geeigneter abstrakter Datentypen hilft hier wesentlich, den räumlichen Zugriff auf raumbezogene Daten (Punktobjekte und räumlich ausgedehnete Objekte) zu vereinfachen. Mit einer entsprechenden Menge geometrischer Operationen wie Überlappung, Enthaltensein, Entfernungsbestimmung lassen sich dann effektiv komplexe geometrische Algorithmen abwickeln.

5. Datenmodelle zur Unterstützung der Wissensrepräsentation

Wir gehen davon aus, daß WR-Konzepte in künftigen WBVS nicht einstufig realisiert werden, sondern daß vielmehr ein mehrschichtiger Abbildungsprozeß erforderlich ist, um sie von ihrer Externspeicherdarstellung aus an der Benutzerschnittstelle verfügbar zu machen [Wi87]. In diesem Systemmodell wird das Datenmodell eine interne Systemschnittstelle beschreiben, auf die aufbauend die WR-Schicht zu implementieren ist. Die entscheidende Frage ist jetzt die nach den Qualitäten des Datenmodells, um ein Spektrum von WR-Verfahren (siehe Kap. 2) angemessen unterstützen zu können.

Im Grundsatz ist dazu jedes der klassischen Datenmodelle geeignet. Jedoch würde eine solche Abbildung recht kompliziert und ineffizient, da - wie in Kap. 4 am Beispiel des Relationenmodells diskutiert - selbst einfache Mechanismen der Abstraktion oder Objektorientierung nicht vorhanden sind. Erwünscht sind Datenmodelle, die besser an die durch das WR-System anzubietenden Konzepte (geringere konzeptuelle Diskrepanz) angepaßt sind. Eine solche "höhere" Datenmodell-Schnittstelle verkörpert mehr Semantik über die zu verwaltenden Objekte und läßt deshalb bei der Realisierung neben effizienten Operationen auch die Gewährleistung mächtiger Integritätsbedingungen zu. Wie bei den klassischen Datenmodellen soll jedoch Anwendungsneutralität gewahrt bleiben.

Es gibt bisher noch kein allgemein akzeptiertes Datenmodell, das die skizzierte Rolle übernehmen könnte. Jedoch werden seit einigen Jahren heftige Forschungsanstrengungen unternommen, um sogenannte Non-Standard-Datenmodelle zu entwickeln. Hier können nur einige Forschungsansätze erwähnt werden, die oft auf Ergänzungen und Erweiterungen des Relationenmodells abzielen.

Eine Art von Objektorientierung für das Relationenmodell wird in [SRG83, St86] vorgeschlagen. Danach lassen sich Attributtypen von Relationen als abstrakte Datentypen (ADT) definieren, für die entsprechende ADT-Operationen in die DB-Sprache zu integrieren sind. Auf diese Weise kann die Darstellung von Attributwerten auf Tupelebene verborgen werden. Über die Idee, auch Attribute von Typ "Prozedur" zuzulassen und als reguläre DB-Objekte zu behandeln, ist es möglich, eine Reihe von Eigenschaften objektorientierter und semantischer Datenmodelle zu simulieren. Beispielsweise kann über eine DB-Prozedur, die Anweisungen der relationalen DB-Sprache enthält, ein komplexes Objekt abgeleitet werden.

Strukturelle Erweiterungen des Relationenmodells zielen in erster Linie auf die Definition hierarchischer Beziehungen zwischen Relationen und den dazugehörigen Operationen ab. XSQL [HL82] reichert das herkömmliche Relationenmodell um spezielle Datentypen "comp-of" und "ref-to" an, mit deren Hilfe hierarchisch strukturierte Objekte statisch definiert werden können. Das NF^2-Relationenmodell [SS86] gestattet die Spezifikation relationenwertiger Attribute gemäß einer im DB-Schema festzulegenden hierarchischen Struktur. Eine erweiterte (rekursive) Relationenalgebra erlaubt den selektiven Zugriff auf allen Hierarchieebenen; als Anfrageergebnis kann eine Menge von hierarchisch strukturierten Tupeln ausgewählt werden.

Eine allgemeinere strukturierte Objektorientierung bieten Modelle, die sich als Erweiterungen des Entity-Relationship-Ansatzes verstehen. Teilweise werden zusätzlich Abstraktionsmechanismen wie Generalisierung und Aggregation verfügbar gemacht, wodurch auch der Name "Semantische Datenmodelle" gerechtfertigt wird [HK87]. Sogenannte molekulare Objekte oder Aggregationen [BB84] können zur strukturellen Definition von Objekten mit netzwerkartigen und rekursiven Typstrukturen herangezogen werden. Das MAD-Modell [Mi88] erlaubt die direkte Spezifikation solcher Objekttypen in der Anfragesprache, was die dynamische Ableitung der zugehörigen Objekte - aus Atomen verschiedenen Typs werden Moleküle zusammengebaut - impliziert. Im [HMM87] wurde nachgewiesen, daß ein mit Abstraktionskonzepten ausgestattetes Frame-Modell sich in einfacher und natürlicher Weise auf das MAD-Modell abbilden läßt.

Wenn auch eine verhaltensmäßige Objektorientierung erreicht werden soll, so wird in der Regel davon ausgegangen, daß diese mit Hilfe eines ADT-Konzeptes oder einer prädikatenlogischen Sprache in einer zusätzlichen Abbildungsschicht erzeugt wird. Die objektbezogenen Operationen werden dann als (Teil der) DB-Sprache verfügbar gemacht.

6. Ein Architekturansatz für Wissensbankverwaltungssysteme

Eine Vielzahl von Forschungsprojekten untersucht momentan verschiedene Ansätze der Integration von Datenmodellen und WR-Verfahren in WBVS oder auch in sogenannten Non-Standard-DBS zur Abwicklung von Ingenieuraufgaben oder Büroarbeiten. Dabei ist die Informations- und Wissensmodellierung nur ein, wenn auch sehr wichtiger Problembereich. Da das Modellierungsvermögen der klassischen Datenmodelle im Vergleich zu den Ansprüchen der WR-Verfahren eher als bescheiden zu charakterisieren ist, wird durchwegs davon ausgegangen, daß "semantischere" Datenmodelle verfügbar sind, um keine zu große konzeptuelle Kluft überbrücken zu müssen (siehe Kap. 5).

Um realistische Aussagen zu gewinnen, müssen alle Fragen der Systemarchitektur und der Anwendungsumgebung geklärt und im Zusammenhang bewertet werden. Schon bei der Entwicklung von DBS war der Aspekt der Informationsmodellierung nur einer von vielen Aufgaben; ihre Bewährung in realen Anwendungssituationen hängt - was hier explizit unterstrichen werden soll - gleichermaßen auch von folgenden Eigenschaften ab:

- effiziente Verwaltung großer Datenmengen
- gemeinsame Nutzung der Daten durch mehrere Benutzer
- hohe Zuverlässigkeit und Verfügbarkeit
- Transaktionskonzept mit Gewährleistung der Datenpersistenz
- Verwaltung verteilter Datenbestände.

Bevor solchen neuartigen Systemen eine Praxistauglichkeit bescheinigt werden kann, sind deshalb noch erhebliche Entwicklungsleistungen zu erbringen. Für die Verwaltung und Handhabung großer Wissensbasen sind nicht nur effiziente Techniken erforderlich, sondern es werden auch hohe Ansprüche an die Robustheit, Flexibilität und Benutzerfreundlichkeit solcher Systeme gestellt.

Diese "erweiterte" Problemstellung legt es nahe, über das Verhältnis von DBVS und WBVS nachzudenken, um gegebenenfalls das Leistungsangebot des einen bei der Realisierung des anderen zu nutzen. Dabei geht es sicher nicht um eine bloße Übernahme eines DBVS und seine lose oder enge Kopplung an eine wissensbasierte Anwendung. Vielmehr ist u.a. aus Leistungsüberlegungen heraus eine integrierte Lösung anzustreben, wobei jedoch auch erhebliche Anpassungen und Änderungen bei heutigen DBS-Architekturen erforderlich werden. In diesem Zusammenhang haben sich drei ausgeprägte Forschungsrichtungen herausgebildet: objektorientierte, erweiterbare und DBS-Kern-Architekturen [Di86].

Die erste Richtung versucht, einen (strengen) Objektbegriff im System zu verkörpern, wobei das Datenmodell jedes Objekt (Entity) der Anwendung durch genau ein DB-Objekt darzustellen gestattet und geeignete Operationen zur Objektmanipulation bereithält oder zu definieren erlaubt. Dieses Ziel läßt sich offensichtlich auf vielfältige Weise annähern wie ein Blick auf die momentane Forschungslandschaft (POSTGRES, GemStone, ORION usw.) zeigt. Beurteilungsmaßstab bei diesen Ansätzen wird sein, wie gut die charakteristischen Eigenschaften der Objektorientierung - nämlich Datenabstraktion, Objektidentität, Typ- und Klassenkonzept sowie Vererbungsmechanismen - in den Systementwurf umgesetzt werden können.

Erweiterbare DBS sollen Vorkehrungen besitzen, daß sie "dynamisch" so erweitert und abgeändert werden können, um die gestellten Anforderungen möglichst gut zu erfüllen. Ehrgeizige Projekte zielen darauf ab, bei Erweiterungen nach Möglichkeit jegliche Codeanpassung zu vermeiden und ein automatisiertes Anpassungsverfahren anzubieten (STARBURST, GENESIS, EXODUS usw.) Beispielsweise sollen nach [Ha88] erweiterbare DBS die Ergänzung der Datenmodell-Schnittstelle mit neuen, problembezogenen Datentypen erlauben, was massive Rückwirkungen auf andere Schichten der DBS-Architektur hat: Es müssen z.B. zugeschnittene Speicherungsstrukturen und Zugriffspfade bereitgestellt werden, der Anfrageoptimierer muß angepaßt werden, um diese Strukturen ausnutzen zu können, Synchronisation und Recovery werfen neue Probleme auf.

Die DBS-Kern-Architektur [HR85, SW87] und ihre Verwendung im Rahmen eines WBVS soll hier als Beispiel etwas genauer diskutiert werden. Je nach Schwerpunktsetzung erlaubt sie die Berücksichtigung von Eigenschaften, die erweiterbare oder objektorientierte DBVS auszeichnen. Die wichtigsten Aspekte dieses Architekturvorschlags sind in Bild 5 veranschaulicht. Er sieht eine strikte Zweiteilung der Gesamtarchitektur in DBS-Kern und Modellabbildung vor, wobei der DBS-Kern das gewählte (Non-Standard-)Datenmodell als wichtigste interne Schnittstelle implementiert.

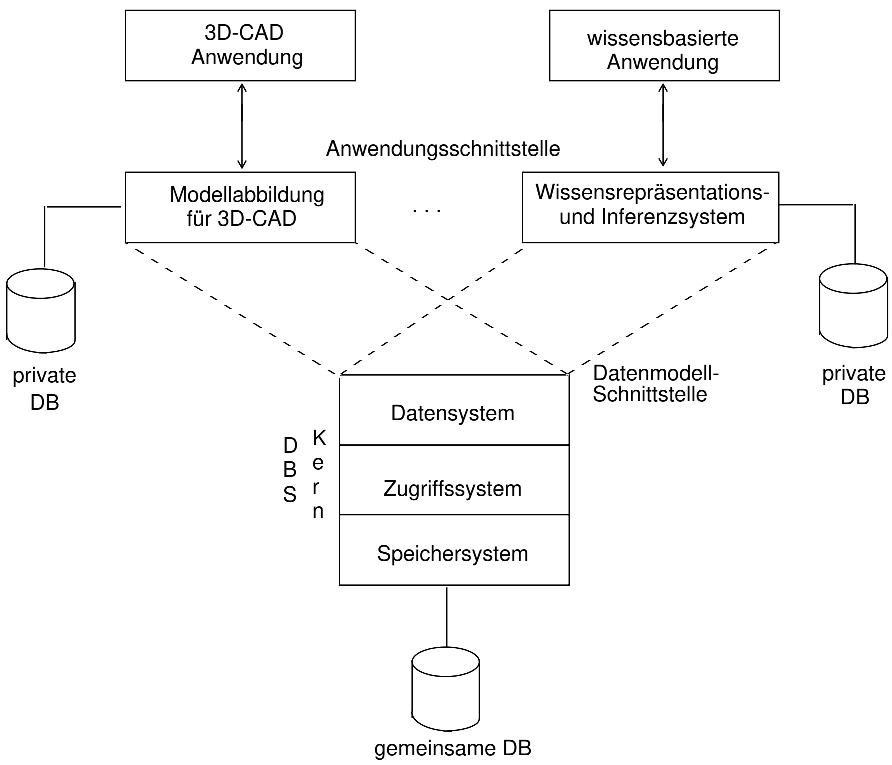

Bild 5: Wissensbankverwaltung im Rahmen einer DBS-Kern-Architektur

Die Modellabbildung als oberste Systemschicht setzt auf der das Datenmodell verkörpernden Schnittstelle auf, die noch keinerlei spezifische Anwendungsorientierung aufweist. In dieser Schicht wird der gewünschte Anwendungsbezug für die an der Anwendungsschnittstelle angebotenen Objekte und Operationen erzielt. Beispielsweise läßt sich hier eine verhaltensmäßige Objektorientierung erreichen, wobei auch die Kontrolle komplexer anwendungsbezogener Integritätsbedingungen zu gewährleisten ist. Weiterhin könnten in dieser Schicht die in Kap. 4.2 und 4.3 skizzierten Konzepte, ggf. auf die bestimmte Anwendung zugeschnitten, realisiert werden. Eine solche Modellabbildung läßt sich relativ leicht und unabhängig vom Restsystem um Operationen (z.B. beim ADT-Konzept) ergänzen; sie gestattet somit ein gewisses Maß an Erweiterbarkeit. Das Konzept der DBS-Kern-Architektur fördert noch eine andere Art der Erweiterbarkeit, da mehrere Modellabbildungen in einem System unterstützt werden können. Man geht davon aus, daß man für jede Anwendungsklasse jeweils eine zugeschnittene (anwendungsunterstützende) Schnittstelle verfügbar macht, wobei ein hoher Grad an Objektorientierung (gemessen an den Anwendungsobjekten) und leistungsfähige Abstraktionskonzepte erreicht werden sollen. Auf diese Weise kann einer Anwendung nach Bedarf ein oder mehrere WR-Verfahren zugänglich gemacht werden. Eine Modellabbildungsschicht kann beispielsweise Operationen für die volumenorientierte 3D-Modellierung anbieten (wobei intern Frames und Abstraktionskonzepte zur Kontrolle komplexer Integritätsbeziehungen eingesetzt werden), eine andere kann wissensbasierte Anwendungen mit einer Mischform aus framebasierten WR-Verfahren und Regelverarbeitung unterstützen.

Der DBS-Kern besteht aus drei hierarchisch angeordneten Schichten, deren Aufgaben sich grob folgendermaßen charakterisieren lassen:

- Das Speichersystem ist für die Verwaltung eines (großen) Systempuffers und für die Abbildung der internen Objektrepräsentationen auf die externen Speichermedien verantwortlich. Dabei nutzt es im allgemeinen die Funktionen eines Dateisystems aus (Betriebssystem-Schnittstelle).
- Das Zugriffsystem ist für die Speicherung und Aktualisierung der internen Objektrepräsentationen sowie für die Effizienz der Suchvorgänge verantwortlich. Dazu hat es ein Spektrum verschiedenartiger Zugriffspfade zur Verfügung zu stellen, um unterschiedlichste Suchanforderungen "zufriedenstellend" bedienen zu können.
- Das Datensystem übernimmt die Übersetzung und Optimierung der Datenmodelloperationen, die mengenorientiert sind und sich auf komplexe Objekte beziehen. Solche mächtigen Operationen müssen mit Hilfe der satzorientierten Operationen des Zugriffsystems ausgeführt werden. Dabei sind neben sequentiellen auch parallele Auswertungsstrategien denkbar.

Die konkrete Ausgestaltung dieser Schichten wird sich von der konventioneller DBVS erheblich unterscheiden. Jedoch besteht noch wenig Klarheit darüber, wie wissensbasierte Anwendungen bei sehr großen Datenmengen zu unterstützen sind. Eine Schlüsselrolle werden hier die verfügbaren Zugriffspfade (Zugriffssystem) spielen, da Vielfalt der gespeicherten Zusammenhänge und weniger restriktive Suchanforderungen wesentlich höhere Ansprüche an die Qualität der Suchalgorithmen stellen. Direkter Schlüsselzugriff und Bereichsanfragen lassen sich noch am ehesten durch bewährte Methoden (B*-Baum, erweiterbares Hashing) abdecken; jedoch gibt es auch hierfür schon Optimierungsvorschläge durch hybride Indexstrukturen [LL87]. In ähnlicher Weise müssen zugeschnittene Verfahren und Strukturen für Verbundoperationen entwickelt werden. Weiterhin sind Mechanismen zur Unterstützung des räumlichen und zeitlichen Wissens vorzusehen; für Aufgaben des mehrdimensionalen Zugriffs wurden in den letzten Jahren eine Reihe interessanter Strukturen wie das GRID-File oder das BANG-File [NHS84, Fr87] vorgeschlagen. Der Zugriff auf große Datenbestände mit Suchmustern verlangt Vorkehrungen, da sequentielle Suche nicht tolerierbar erscheint. Hier könnten beispielsweise Signaturdateien [Fa85], wie sie für das Text Retrieval entworfen worden sind, eingesetzt werden. Erheblich größere

Probleme scheinen die "freie" Suche [Hä88] oder der inhaltsorientierte Zugriff auf benutzerdefinierte Attribute zu verursachen. Im ersten Fall kann der Suchraum wegen fehlender oder durch Prädikate spezifizierte Typ-/Klassenangaben nicht oder nicht effektiv eingeschränkt werden. Im zweiten Fall kann kein Zugriffspfad (im herkömmlichen Sinn) angelegt werden, da in jedem Objekt andere Attribute spezifiziert sein können. Neben dem Problem der Sucheffizienz und der Optimierung der betreffenden Anfragen gibt es sicher noch eine Vielfalt von Fragen, die den Aufbau des DBS-Kerns betreffen, zu klären. Hier sollen nur stichwortartig noch einige wichtige Aspekte genannt werden:

- Replikation und Einsatz von Redundanz auf allen Ebenen kann sich als sehr hilfreich für die Leistungsaspekte erweisen.
- Regelverarbeitung, d.h. die Behandlung von implizitem und explizitem Wissen, sowie Mechanismen zur Unterstützung aktiver DBS sind effizient zu integrieren.
- Umgang mit vagem oder nicht vorhandenem Wissen (wie schon bei der Suche angedeutet) sollte möglich sein.

Ein wesentlicher Vorteil der strikten Zweiteilung der Architektur wird sichtbar, wenn ein Mehrbenutzersystem auf einem Netz von Arbeitsplatzrechnern (was künftig in vielen anspruchsvollen Anwendungen eine Standardumgebung sein wird) realisiert werden soll. Der DBS-Kern kann einem Server-Komplex zugeordnet werden, während die Anwendungen zusammen mit der Modellabbildungsschicht auf einzelnen Workstations ablaufen. Aus Leistungsgründen muß die Kommunikation zwischen Server und Workstation minimiert werden. Hohe Lokalität der Verarbeitung läßt sich durch Extraktion der benötigten DB-Daten und ihre Zwischenpufferung für die Dauer der Verarbeitung in der Workstation (Working Memory/Objektpuffer) gewährleisten. Voraussetzung ist ein angepaßtes Verarbeitungsmodell, das die ausschließliche lokale Verarbeitung einer Anwendung fördert. Dazu sind zugeschnittene Transaktionsmodelle zu entwickeln, die auch bei den speziellen Charakteristika der Systemumgebung und des Anwendungsspektrums eine möglichst reibungslose Nutzung der gemeinsamen DB gestatten. Physische und logische Entkopplung, unterstützt durch private DB's und Transaktionskonzept, erzielen auch einen hohen Grad an Fehlerisolation zwischen Workstation/Anwendung und Server/DBS-Kern. Eine ausführlichere Diskussion aller Vorteile dieser Architektur findet sich in [Hä89].

7. Ausblick

Oft wird die Hypothese aufgestellt, daß die Effizienz heutiger DBS gerade in der Beschränkung der Modellierungsmöglichkeiten klassischer Datenmodelle begründet liegt. Die Reichhaltigkeit der Modellierungskonzepte von WR-Verfahren würde demnach erhebliche Leistungseinbußen implizieren. Gerade diese Konsequenz sollte durch angepaßte Architekturkonzepte, Lokalitätserhaltung, Zugriffspfadunterstützung usw. vermieden bzw. gemildert werden. Auf der Suche nach solchen Lösungen stehen wir jedoch noch ganz am Anfang. Einen Forschungsansatz stellt das KRISYS-Projekt [Ma88b] dar, in dem zur Zeit ein Prototyp eines WBVS realisiert wird. Als Datenmodell des DBS-Kerns wurde das MAD-Modell implementiert; die Modellabbildung baut ein mit den vier allgemeinen Abstraktionskonzepten angereichertes Frame-Modell auf. Schließlich verkörpert die Benutzerschnittstelle ein hohes Maß an "Wissensrepräsentationsunabhängigkeit" - durch Einsatz eines Prädikatenkalküls erster Ordnung (mit ASK/TELL).

Danksagung

Ich möchte mich bei N. Mattos und B. Mitschang für kritische Diskussionen zum Thema und für das Korrekturlesen des Manuskripts bedanken.

8. Literaturverzeichnis

[Ba85] Bayer, R.: Database Technology for Expert Systems, in: Wissensbasierte Systeme, IFB 112, Springer Verlag, 1985, S. 1-16.

[BB84] Batory, D.S., Buchmann, A.P.: Molecular Objekts, Abstract Data Types and Data Models: A Framework, in: Proc. 10th VLDB Conf., Singapore, 1984, pp. 172-184.

[Br83] Brachman, R.J.: What IS-A Is and Isn't: An Analysis of Taxonomic Links in Semantic Networks, in: IEEE Computer, Vol. 16, No. 10, Oct. 1983, pp. 30-36.

[Ch80] Chamberlin, D.D.: A summary of user experience with the SQL data sublanguage, in: Deen, S.M., Hammersley, P. (eds.): Int. Conf. on Data Bases, Heydon, London, 1980, pp. 181-203.

[[Ch76] Chen, P.P.: The Entity-Relationship-Model - Toward a Unified View of Data, in: ACM TODS, Vol. 1, No. 1, 1976, pp. 9-36.

[Da83] Date, C.J.: An Introduction to Database Systems, Addison Wesley, 1983 (3. Auflage).

[Di86] Dittrich, K.R.: Object-Oriented Database Systems - A Workshop Report, in: Proc 5th Int. Conference on Entity-Relationship Approach, North Holland Publ. Comp., 1986.

[Fa85] Faloutsos, C.: Access Methods for Text, in: Computing Surveys, Vol. 17, No. 1, March 1985, pp. 49-74.

[FK85] Fikes, R., Kehler, T.: The Role of Frame-based Representation in Reasoning, in: Communications of the ACM, Vol. 28, No. 9, Sept. 1985, S. 904-920.

[Fr87] Freeston, M.W.: The BANG file: a new kind of grid file, in: Proc. ACM SIGMOD Conf., San Francisco, 1987.

[Ha88] Haas, L.M. et. al.: An Extensible Processor for an Extended Relational Query Language, IBM Research Report, RJ 6182, San Jose, Calif., April 1988.

[Hä88] Härder, T.: Eine DB-gestützte Architektur für Wissensbankverwaltungssysteme, Bericht des ZRI 4/88, Universität Kaiserslautern, 1988.

[Hä89] Härder, T.: Non-Standard DBMS for Support of Emerging Applications - Requirement Analysis and Architectural Concepts, in: Proc. HICSS-22, Hawaii, Jan. 1989, pp. 549-558..

[HL82] Haskin, R.L., Lorie, R.A.: On extending the functions of a relational database system, Proc. ACM SIGMOD Conf., 1982, pp. 207-212.

[HMM87] Härder, T., Mattos, N., Mitschang, B.: Abbildung von Frames auf neuere Datenmodelle, in: Proc. "German Workshop on Artificial Intelligence" GWAI'87, IFB 152, Springer Verlag, 1987, S. 396-405.

[HK87] Hull, R., King, R.: Semantic Database Modelling: Survers, Applications, and Research Issues, in: ACM Computing Survers, Vol. 19, No. 3, Sept. 1987, pp. 201-260.

[HR85] Härder, T., Reuter, A.: Architektur von Datenbanksystemen für Non-Standard-Anwendungen, Informatik-Fachberichte 94, Berlin, Heidelberg, New York, Tokyo: Springer Verlag, 1985, pp. 253-286.

[JK83] Jarke, M., Vassilion, Y.: Coupling Expert Systems with Database Management Systems, in: Artificial Intelligence Applications for Business, Ablex Publ. Comp., New Jersey, 1983, pp. 65-85.

[Kä88] Käfer, W.: Ein Modell zur Integration der Zeit in relationalen Datenbanksystemen, Forschungsbericht 27/88 des SFB 124, Universität Kaiserslautern, 1988.

[Ke85] Kerschberg, L. et. al.: Expert Database Systems (Workshop Review), in: Proc. 1st Int. Workshop on Expert Database Systems, ACM Publication, 1985, pp. 414-417.

[Kl83] Klopprogge, M.R.: Gegenstands- und Beziehungsgeschichten: Ein Konzept zur Beschreibung und Verwaltung zeitveränderlicher Informationen in Datenbanken, Dissertation, Karlsruhe, 1983.

[LL87] Litwin, W., Lomet, D.V.: A New Method for Fast Data Searches with Keys, in: IEEE Software, March 1987, pp. 16-24.

[Ma88a] Mattos, N.M.: Abstraction Concepts: the Basis for Data and Knowledge Modelling, in: Proc. 7th Int. Conf. on Entity-Relationship Approach, Rome, Nov. 1988.

[Ma88b] Mattos, N.M.: KRISYS - A Multi-layered Prototype KBMS Supporting Knowledge Independence, in: Proc. Int. Computer Science Conf., Hongkong, Dec. 1988.

[Mi88] Mitschang, B.: Ein Molekül-Atom-Datenmodell für Non-Standard-Anwendungen - Anwendungsanalyse, Datenmodellentwurf, Implementierung - Dissertation, Universität Kaiserslautern, März 1988.

[My80] Mylopoulos, J.: An Overview of Knowledge Representation, in: Proc. Workshop on Data Abstraction, Databases and Conceptual Modelling, Pingree Park, Colorado, June 1980.

[NHS84] Nievergelt, J., Hintenberger, H., Sevcik, K.C.: The Grid File: an adaptable, symmetric multikey file structure, in: ACM TODS, Vol. 9, No. 1, March 1984, pp. 38-71.

[Re87] Reuter, A.: Kopplung von Datenbank- und Expertensystemen, in: Informationstechnik-Computer, Systeme, Anwendungen, 29. Jg., Heft 3, 1987, S. 164-175.

[Ro87] Rosenthal, A. et. al.: Query Facilities for Part Hierarchies: Graph Traversal, Spatial Data and Knowledged-Based Detail Suppression, Research Report, CCA, Cambridge, MA., 1987.

[RV82] Requicha, A.A.G., Voelcker, H.B.: Solid modelling: A historical summary and contemporary assessment, IEEE Computer Graphics Appl. 2, 1982, pp. 9-24.

[Sn87] Snodgrass, R.: The Temporal Query Language TQuel, ACM TODS, Vol 12, No. 2, June 1987, pp. 247-298.

[SRG83] Stonebraker, M., Rubenstein, B., Guttman, A.: Application of abstract data types and abstract indices to CAD databases, Proc. Database Week - Engineering Design Applications, IEEE Comp. Soc., 1983.

[SS86] Schek, H.-J., Scholl, M.H.: The Relational Model with Relation-Valued Attributes, in: Information Systems, Vol. 11, No. 2, 1986.

[SS77] Smith, J.M., Smith, D.C.P.: Database abstractions: Aggregation and Generalization, ACM Trans. Database Syst. 2, 1977, pp. 105-133.

[St86] Stonebraker, M.: Inclusion of New Types in Relation Database Systems, in: Proc. International Conference on Data Engineering, Los Angeles, 1986, S. 262-269.

[SW87] Schek, H.-J., Weikum, G.: DASDBS: Konzepte und Architektur eines neuartigen Datenbanksystems, Informatik Forsch. Entw. 2, 1987, S. 105-121.

[Wi87] Wiederhold, G. et. al.: Architectural Concepts for Large Knowledge Bases, in: Proc. "German Workshop on Artificial Intelligence" GWAI'87, IFB 152, Springer Verlag, 1987, S. 366-385.

[Wo83] Woods, W.A.: What's Important About Knowledge Representation?, in: IEEE Computer, Vol. 16, No. 10, Oct. 1983, pp. 22-27.