Charakteristiken des Komplex-Objekt-Begriffs und Ansätze zu dessen Realisierung

Bernhard Mitschang
Universität Kaiserslautern, Fachbereich Informatik
Erwin-Schrödinger-Straße
D-6750 Kaiserslautern

Überblick:

Die Modellierung und Verwaltung von komplexen Objekten stellt eines der Hauptprobleme für den effektiven Einsatz von Datenbanksystemen in den sog. nichtkonventionellen Anwendungsgebieten dar. Unter dem Begriff "Komplex-Objekt" wird dabei die Sicht der Anwendung auf den betreffenden Realitätsausschnitt verstanden (etwa Entwurfs- und Konstruktionsobjekte im CAD/CAM/CAE-Bereich, Objekte in der Computergrafik, Texte in der Büroautomatisierung etc.).
Für eine adäquate Modellierung bzw. zur Konzeption von geeigneten Datenmodellen müssen die Anforderungen der Komplex-Objekt-Sicht erkannt und aufgegriffen werden. Hier werden nun zum einen die obigen Anforderungen in Form von inhärenten Charakteristiken dieser Komplex-Objekte ermittelt und anhand einleuchtender Beispiele klar herausgearbeitet. Zum anderen werden bestehende Lösungsansätze in dem vorgegebenen Rahmen erörtert sowie erste Hinweise zum Entwurf von auf den Komplex-Objekt-Begriff zugeschnittenen Datenmodellen angegeben.

Abstract:

One of the main problems encountered when using database systems for non-standard applications is modeling and managing of complex objects. The notation "complex object" encompasses the application's view of the interesting part of the real world (for instance design objects in CAD/CAM/CAE or computer graphic, text objects in office automation, etc.). For adequate modeling and also for the development of proper data models, one has to realize and to comprise the requirements of the complex object view.
This work analyses the above requirements as inherent complex object characteristics. Subsequently some existing solutions to the complex object problem are considered and lastly a few general hints for complex object data models are given.

1. Einleitung

Die stetige Erweiterung und Vergrößerung von rechnergestützten Anwendungen in den Spezialbereichen wie z.B. Prozeßdatenverarbeitung, geographische Datenhaltung, Bild- und Textverarbeitung sowie CAD/CAM/CAE erfordern in wachsendem Maße Datenbankunterstützung. Die Brauchbarkeit herkömmlicher Datenbanksysteme (DBS) für einen Einsatz in solchen Non-Standard-Anwendungsgebieten wird aber zunehmend in

Frage gestellt (/Ea80/, /GP83/, /Lo81/, /Lo83/, /Si80/). Die wichtigsten Problempunkte umfassen adäquate Modellierungsmöglichkeiten, bessere Datentypunterstützung, anwendungsgerechte (Zugriffs-)Operationen, Erhaltung der Datenintegrität, angepaßtes Transaktionskonzept und das insgesamt extrem schlechte Leistungsverhalten.

Schaut man sich nun obige Problembereiche im Umfeld der Anwendungen genauer an, so erkennt man (siehe auch /PS84/ Seite 5), daß das eigentliche, zentrale Problem in der anwendungsgerechten Modellierung und Verwaltung der Anwendungsobjekte liegt. Unter dem Begriff (Anwendungs-)Objekt wird hierbei die Sicht der Anwendung auf den betreffenden Realitätsausschnitt subsumiert. Diese "Sicht" umfaßt deshalb sowohl die objektbeschreibenden Daten selbst, als auch die korrekten, konsistenten Operationen auf dem Objekt bzw. auf dessen Beschreibungsdaten. Dabei spielt es prinzipiell keine Rolle, ob man es mit den

- aus verschiedenen dreidimensionalen Teilobjekten zusammengesetzten Entwurfs- und Konstruktionsobjekten im Anwendungsgebiet CAD/CAM/CAE, mit den
- punkt-, linien- und flächenartigen Objekten (Straßenkreuzung, Wasserleitung, Grundstück) in der geographischen Datenverarbeitung oder aber mit den
- langen Texten in der Büroautomatisierung bzw. mit den matrixartigen Bildern bei der Bildverarbeitung zu tun hat.

Aus diesem Grunde befaßt sich das nachfolgende Kapitel noch etwas genauer mit dem intuitiv schon gebildeten "Komplex-Objekt"-Begriff, bevor in Kapitel 3 eine mehr formalisierte und verallgemeinerte Vorstellung des Begriffs aufgezeigt wird. Die dort eingeführten Charakteristiken des Komplex-Objekts repräsentieren gleichzeitig eine Anforderungsliste für geeignete Datenmodelle. In diesem Sinne werden dann in Kapitel 4 einige existierende oder geplante Realisierungskonzepte für Komplex-Objekte vorgestellt und eine Zielvorstellung erarbeitet, die in einige vielversprechende und weiterführende Lösungsansätze mündet. Zum Schluß wird, anschließend an eine kurze Zusammenfassung der Ergebnisse, noch ein Ausblick auf weitere Arbeiten gegeben.

2. Einführung des Komplex-Objekt-Begriffs

Als Grundlage für die nachfolgenden Diskussionen und Überlegungen und auch als Beschreibung unseres Verständnisses von Komplex-Objekten, werden im folgenden einige anschauliche Beispiele aus verschiedenen Non-Standard-Anwendungsbereichen angegeben.

Beispiel 1: Komplex-Objekte in Landinformationssystemen (/Fr83/, /GP83/)

Die Daten von Landinformationssystemen der geographischen Datenhaltung (siehe Abbildung 1) lassen sich unterteilen in nicht-geometrische und geometrische Sachverhalte. Dabei stellen die nicht-geometrischen Sachverhalte herkömmliche, einfach strukturierte Daten dar (z.B. Personalinformationen, Nutzungsdaten,

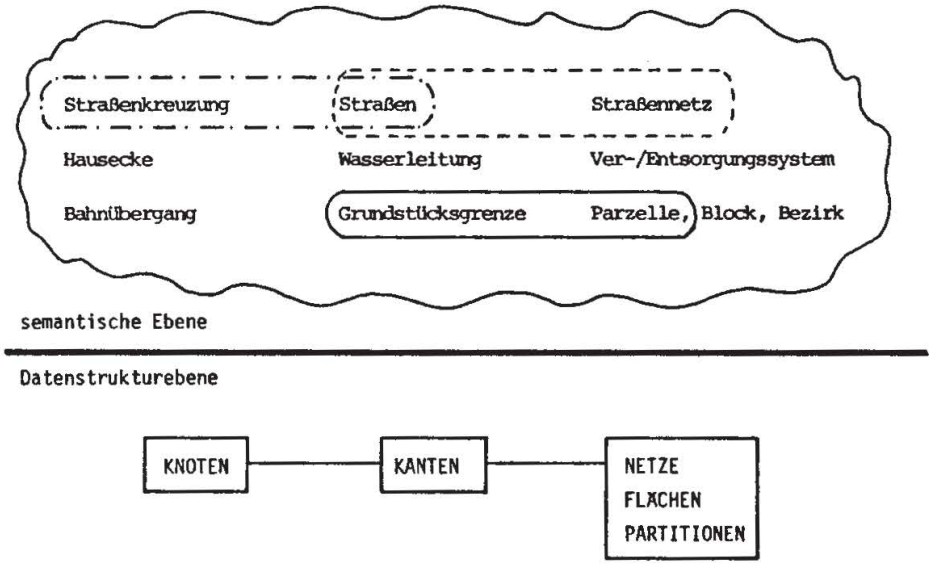


Abbildung 1: Datenausschnitt eines geographischen Informationssystems zur Stadtplanung

statistische Daten), die mit konventionellen Datenmodellen existierender DBS ausreichend zu beschreiben und zu verwalten sind. Hingegen repräsentieren die geometrischen Daten fast ausschließlich graphische, in Plänen und Karten darzustellende Sachverhalte, die nur schwierig und umständlich zu handhaben sind. Die wichtigsten dieser geometrischen Daten sind Knoten, Kanten, Flächen(-Partitionen) und Netze.

KNOTEN und KANTEN sind aus Benutzer- oder Anwendungssicht keine selbständigen Objekte, sondern nur Komponenten der komplexeren Objekttypen FLAECHE und NETZ. Die Knotenobjekte sind extrem einfach und tragen die metrische Information als Koordinatenwerte. Die Kantenobjekte bestimmen die Form der Linien und sind aus Knotenobjekten zusammengesetzt. Flächenobjekte sind entweder aus Kantenobjekten oder direkt aus Knotenobjekten aufgebaut, wohingegen Netzobjekte sichtbar aus einzelnen Kantenobjekten zusammengesetzt sind. Eine (Flächen-)PARTITION ist die Einteilung einer Gesamtfläche in Teilfächen, so daß sich keine zwei Teilflächen überdecken und alle Teilflächen zusammen genau wieder die Gesamtfläche ergeben. Die Partition ist also ebenfalls von komplexem Aufbau und besteht definitionsgemäß aus Flächenobjekten.

Beispiel 2: Komplex-Objekte im Schaltungsentwurf (/DK84/, /Ka83/, /Ne83/)

Der VLSI-Chip-Entwerfer sieht einen elektronischen Schaltkreis in verschiedenen Repräsentationen, z.B. in einer funktionalen Spezifikation, als Schaltkreisdiagramm oder als Layout (siehe Abbildung 2). Jede Repräsentation ist einer unterschiedlichen Entwurfsphase angepaßt und zugeordnet. Es muß immer sichergestellt sein, daß die verschiedenen Repräsentationen auch das gleiche Entwurfsobjekt beschreiben. Da der Entwurfsprozeß als versuchend, vorläufig und provisorisch anzusehen ist, gibt

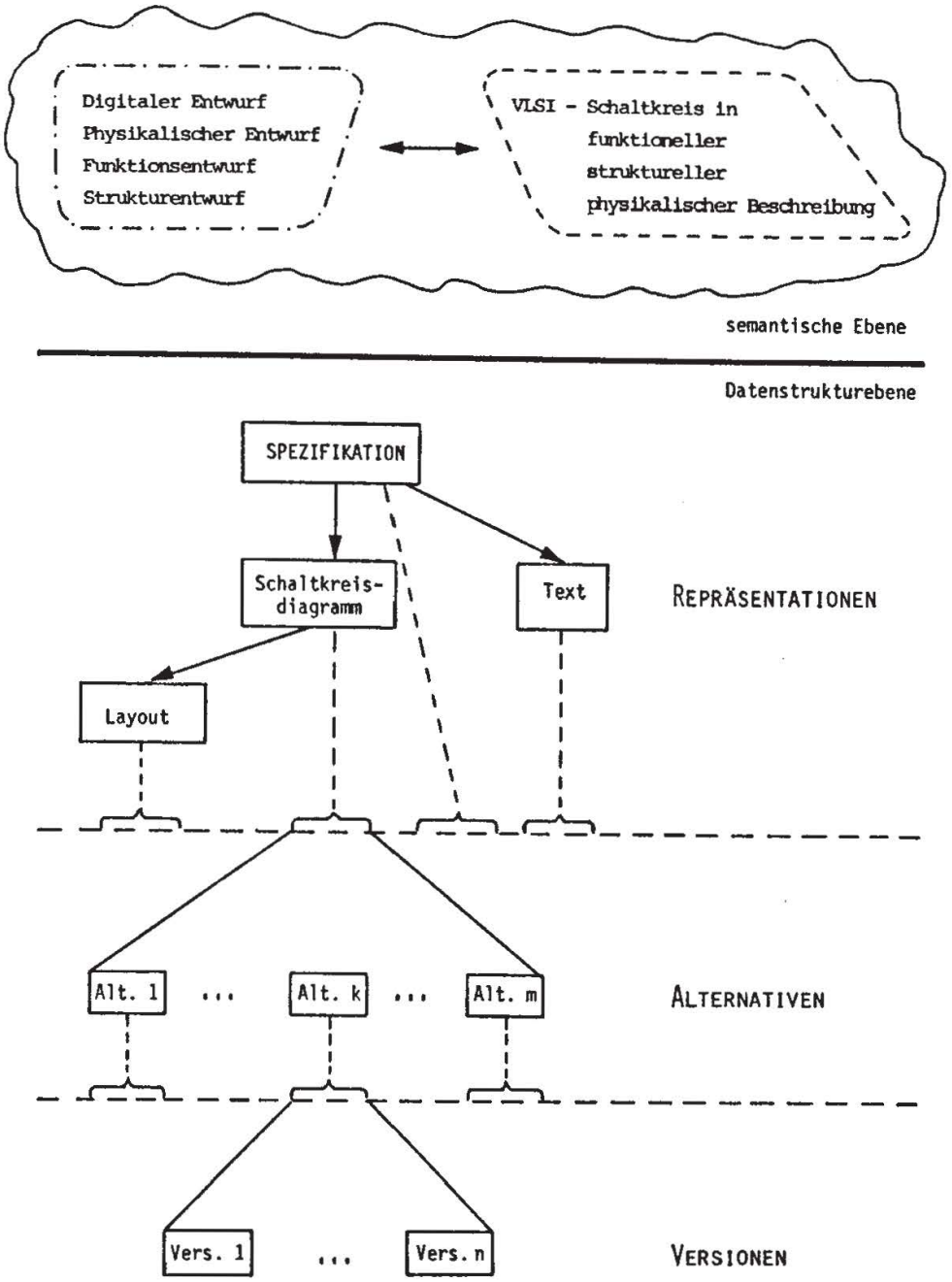


Abbildung 2: Datenausschnitt im VLSI-Schaltungsentwurf

es Alternativen und Versionen unterhalb der Repräsentationsebene. Die Versionen erlauben einen Einblick in die frühere und momentane Entwurfsabsicht und stellen oft Verbesserungen und Korrekturen am Entwurfsobjekt dar. Hingegen ermöglichen die Alternativen ein Experimentieren mit verschiedenen Realisierungen der gleichen Funktion. Im Gegensatz zu den Repräsentationen, die für die Anwendung von vornherein feststehen, sind die Alternativen und Versionen variabel, wobei zusätzlich die Alternativen optional und die Versionen zeitlich geordnet sind. Am Ende des Entwurfsprozesses wird die favorisierte Lösung, auch Konfiguration genannt, durch Auswählen der besten Alternativen und Versionen festgelegt.

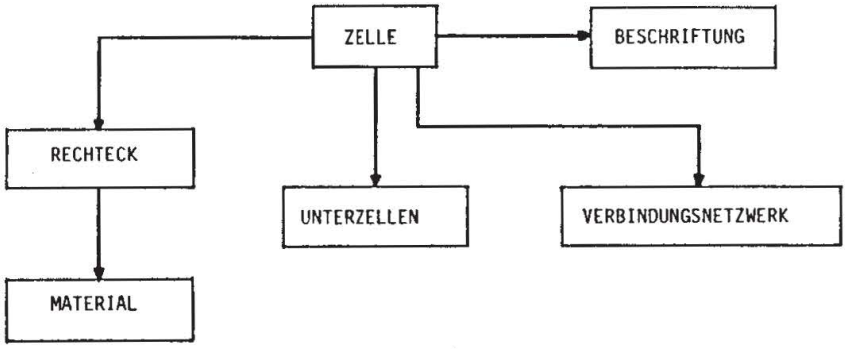


Abbildung 3: Strukturierung innerhalb der Repräsentation LAYOUT

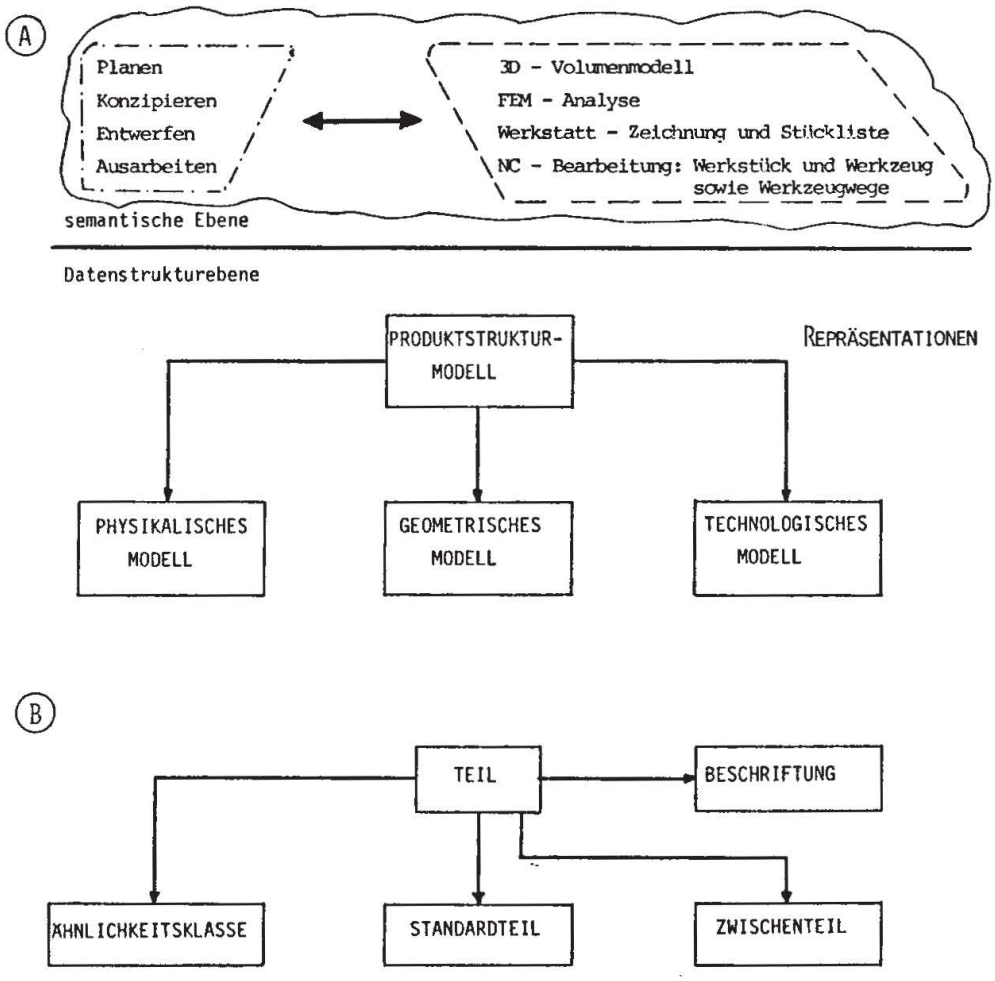


Abbildung 4: a) Datenausschnitt aus dem Maschinenbau
b) Interne Strukturierung des PRODUKTSTRUKTURMODELLS

Zusätzlich zu dieser komplizierten Entwurfsstruktur, bestehend aus den Repräsentationen, Alternativen und Versionen, kommt noch eine komplexe Strukturierung innerhalb der Repräsentationen hinzu. Abbildung 3 stellt diesen Sachverhalt beispielhaft für das Layout dar. Dabei beinhaltet der Objekttyp RECHTECK und MATERIAL die geometrische Information, wohingegen der Objekttyp VERBINDUNGSNETZWERK die topologische Information bezüglich der Nachbarzellen enthält. Zur näheren Beschreibung der Zelle dient der BESCHRIFTUNGS-Objekttyp. Jede Zelle kann wiederum aus (Unter-)Zellen - Objekttyp UNTERZELLEN - aufgebaut sein, die natürlich analog oben dargestellt sind. Dieser Zellenhierarchie entspricht eine Funktionshierarchie innerhalb der Repräsentation Spezifikation und eine sogenannte Strukturhierarchie innerhalb der Repräsentation Schaltkreisdiagramm. Dadurch, daß sich die einzelnen Hierarchieebenen nicht immer einander eindeutig zuordnen lassen, werden die Beziehungen zwischen den verschiedenen Repräsentationen und damit auch der gesamten Entwurfsstruktur zusätzlich verkompliziert•

Beispiel 3: Komplex-Objekte im Maschinenbau (/Eb83/, /EW81/)

Im Anwendungsgebiet rechnergestütztes Entwerfen und Konstruieren im Maschinenbau entsprechen die Entwurfsobjekte dreidimensionalen Körpern. Gemäß der physikalischen, geometrischen, technologischen und strukturellen Eigeschaften dieser technischen Objekte kann man die Daten in einzelne Teilbereiche, die Repräsentationen, aufspalten (siehe Abbildung 4a). So umfaßt der physikalische Bereich etwa die Materialeigenschaften, Massen- und Volumeneigenschaften usw., während im geometrischen Bereich die Geometrie und Topologie der technischen Objekte sowie deren Zeichnungsinformation verwaltet wird. Der technologische Teil enthält Werkzeug- und Maschineninformationen, Bearbeitungshinweise oder NC-Programme, Toleranzinformation etc. Im strukturellen Teilbereich wird die Objektstruktur festgehalten, wie z.B. das Wissen über den Aufbau eines technischen Objekts (siehe Abbildung 4b). Ein dreidimensionaler Körper besteht aus einzelnen ZWISCHENTEILen, die ihrerseits aus TEILen zusammengesetzt sind, welche wiederum entweder ZWISCHENTEILe oder STANDARDTEILe repräsentieren. Den Informationsgehalt der Repräsentation Produktstrukturmodell kan man zum einen noch durch Hinzufügen von BESCHRIFTUNGS- und Bemaßungsangaben erweitern. Zum anderen können zusätzlich Informationen über Produktvarianten und ÄHNLICHKEITSKLASSEn, die dann Teilefamilien bilden, beigefügt werden. Aus Übersichtlichkeitsgründen fehlen hier die Alternativen- und Versionenebenen, die, analog zur Abbildung 2, ebenfalls unterhalb der Repräsentationsebene anzusiedeln sind

In den Bereichen Bildverarbeitung und Büroautomatisierung kommt zusätzlich zu den u.U. schon komplex zusammengesetzten Bild- bzw. Textobjekten (überlagerte Teilbilder oder einzelne Kapitel eines Artikels) noch deren langer und unstrukturierter Datentyp (im MByte-Bereich) hinzu.

Betrachtet man obige Beispiele etwas genauer, so erkennt man, daß in den verschiedenen Anwendungsgebieten jeweils verschiedene Interpretationen/Ansichten des Komplex-Objekt-Begriffs vorherrschen:
- der (rekursiv) aus einfacheren (Teil-)Objekten zusammengesetzte Komplex-

Objekttyp (siehe Beispiel 1)
- der aufgrund des Entwurfsprozesses entstandene Komplex-Objekttyp (siehe Beispiel 2) und
- der wegen seines besonderen Datentyps so benannte Komplex-Objekttyp (siehe Bild- und Textverarbeitung).

Es können aber auch, wie oben in Beispiel 2 und auch in Beispiel 3 angegeben, Mischformen entstehen, die zu einer weiteren Komplexitätserhöhung führen.

Diese vielfältigen Nuancen des Komplex-Objekt-Begriffs verursachen zum einen eine Begriffsüberladung und zum anderen eine extreme Begriffsvielfalt, die eine wissenschaftliche Bearbeitung der bereits in Kapitel 1 angesprochenen Komplex-Objekt-Problematik sehr behindert. Aus diesem Grunde sollen nun im nächsten Kapitel die charakteristischen Eigenschaften der Komplex-Objekte ermittelt und als geeignete Klassifikationskriterien zu einer schärferen Begriffsbildung benutzt werden.

3. Charakteristiken des Komplex-Objekt-Begriffs

In diesem Kapitel wird eine gegenüber dem vorigen Kapitel stärkere Formalisierung des Komplex-Objekt-Begriffs in dem Maße durchgeführt, daß dessen inhärente Eigenschaften deutlich zum Vorschein kommen. Des weiteren werden auch die mit der Einführung von Komplex-Objekten direkt zusammenhängenden Auswirkungen operationaler und konsistenzbezogener Art kurz angesprochen, bevor im vierten Kapitel Realisierungskonzepte für Komplex-Objekte vorgestellt und untersucht werden.

Schaut man sich die Beispiele in Kapitel 2 nochmals genauer an, so erkennt man zum einen, daß die Komplex-Objekte nicht nur (evtl. auch rekursiv) "verschachtelte Strukturen", sondern auch Strukturen zum Beschreiben der gewünschten Entwurfsmethodologie enthalten. Ein Beispiel für eine verschachtelte Struktur ist etwa das folgende: Ein Partitionsobjekt aus Kapitel 2 Beispiel 1 besteht aus einzelnen Flächenobjekten, von denen jedes aus den berandenden Kantenobjekten zusammengesetzt ist. Jedes Kantenobjekt wiederum ist aus Knotenobjekten aufgebaut. Ein VLSI-Chip beispielsweise besteht aus einzelnen Funktionseinheiten, die ihrerseits aus Zellen zusammengesetzt sind, welche wiederum aus Gattern und Transistoren bestehen.
Die im vorigen Kapitel in Beispiel 2 (und auch in Beispiel 3) eingeführten Strukturen Repräsentation, Alternative und Version erlauben den gesamten Entwurfsprozeß inklusive aller Entwurfsphasen, d.h. also die gesamte Entwurfsmethode, besser zu beschreiben. Diese Strukturen (siehe Abbildug 2) werden deshalb auch im folgenden Entwurfsinformationsstrukturen genannt.
Zum anderen ist in dem detaillierteren Beispiel 1 deutlich sichtbar geworden, daß es unterschiedliche Typen von Verschachtelungsstrukturen gibt. Beispielsweise ist es für den Flächenobjekttyp irrelevant, ob dieser direkt aus dem Knotenobjekttyp oder indirekt über den Kantenobjekttyp modelliert wird. Hingegen muß der Netzobjekttyp aus dem Kantenobjekttyp aufgebaut werden. In diesem Fall muß gegenüber dem vorherigen die Verschachtelungsstruktur des Komplex-Objekts sichtbar sein.

Tabelle 1

(A) Charakteristika der (Komplex-)Objekte aus Kapitel 2 Beispiel 1:

KNOTEN - Objekttyp	atomar (Normalobjekt), nicht-eigenständig
KANTAN - Objekttyp	komplex (Komplexobjekt), sichtbar, nicht-eigenständig, aber mehrfachbenutzbar
FLÄCHEN - Objekttyp	komplex (Primitivobjekt), nicht-sichtbar, eigenständig
PARTITION - Objekttyp	komplex (Komplexobjekt), sichtbar, eigenständig
NETZ - Objekttyp	komplex (Komplexobjekt), sichtbar, eigenständig

(B) Beschreibung obiger (Komplex-)Objekte inklusive interner Objektstruktur:

```
KNOTEN    beschreibung   KNO_NUMMER,
                         X_KOORDINATE,
                         Y_KOORDINATE;

KANTEN    beschreibung   KAN_NUMMER,
                            :
          struktur       ENDPUNKTE: LIST(KNOTEN);

FLÄCHEN   beschreibung   FLA_NUMMER,
                            :
          struktur       BERANDUNG: LIST(KNOTEN);
        or
          struktur       BERANDUNG: SET(KANTEN);

NETZ      beschreibung   NRT_NUMMER,
                            :
          struktur       LINIEN:    LIST(KANTEN);

PARTITION beschreibung   PAR_NUMMER,
                            :
          struktur       PARZELLEN: SET(FLÄCHEN);
```

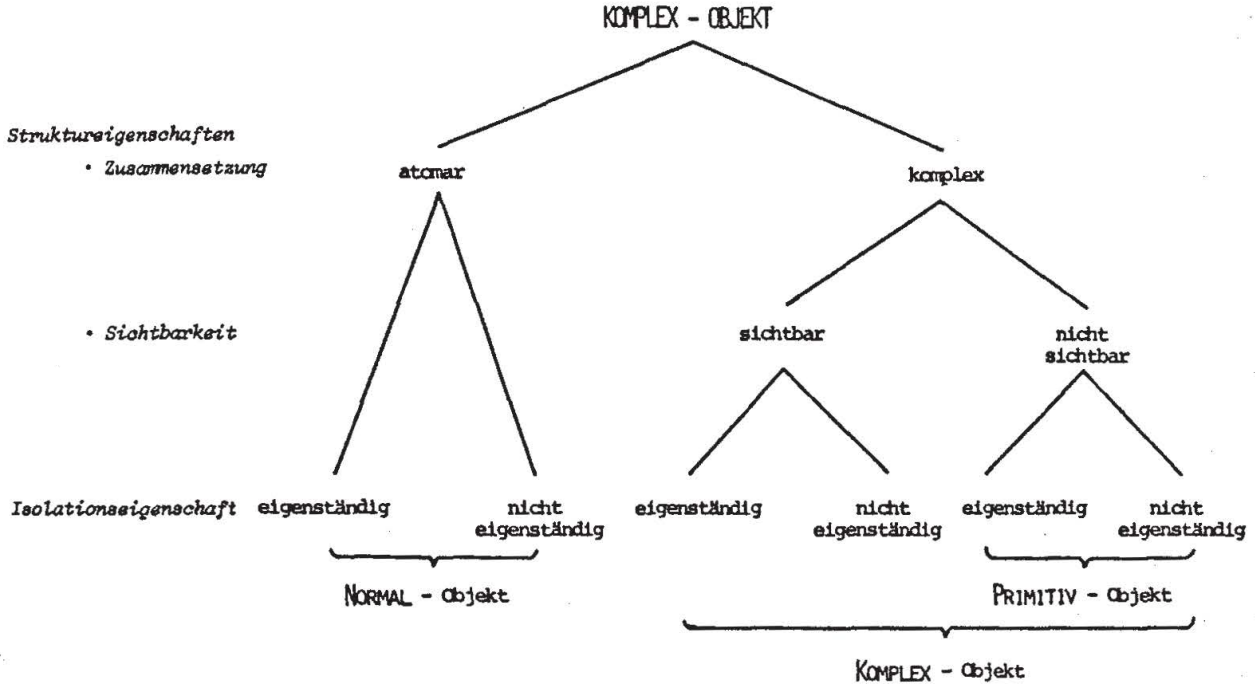


Abbildung 5: Einige Charakteristika von Komplex-Objekten

Abstrahiert man nun von den in dieser Kurzanalyse erkannten Beobachtungen, so gelangt man direkt zu den in Abbildung 5 enthaltenen wichtigen Eigenschaften von Komplex-Objekten. Die ebenfalls charakterisierenden Entwurfsinformationsstrukturen wurden aus Abbildung 5 aus Übersichtlichkeitsgründen weggelassen. Im folgenden werden diese wichtigen Komplex-Objekt-Eigenschaften detaillierter erörtert. Dazu wird ständig auf den vorigen Absatz und auf Tabelle 1 bzw. auf deren umgangssprachliche Beschreibung in Kapitel 2 Beispiel 1 bezug genommen.

Die Struktureigenschaft beschreibt den Aufbau und die Sichtbarkeit eines Komplex-Objekts bzw. eines zugehörigen Objekttyps und teilt sich in die Zusammensetzungs- und die Sichtbarkeitseigenschaft auf. In der Zusammensetzungseigenschaft wird der Aufbau des Komplex-Objekts, d.h. die eigentliche Strukturinformation, angegeben (siehe Tabelle 1b). Man unterscheidet dabei komplexe Objekte, die eine Verschachtelungsstruktur - im folgenden interne Objektstruktur genannt - besitzen, von den atomaren Objekten, die keine solche Struktur aufweisen, allerdings analog den konventionellen Objekttypen durch Attributbeschreibungen näher spezifiziert sind. Die interne Objektstruktur besteht nun aus Komponenten, die wiederum atomare oder komplexe Objekttypen repräsentieren können. Damit ist auch eine rekursive Komponentenbeziehung möglich. In diesem Zusammenhang spricht man häufig vom Komponentenobjekttyp im Gegensatz zum Gesamtobjekttyp und von einer übergeordneten bzw. einer untergeordneten internen Struktur (oder von einem Super- bzw. Subgraph).

Die Sichtbarkeitseigenschaft bezieht sich auf die Sichtbarkeit bzw. Nicht-Sichtbarkeit der internen Objektstruktur und regelt somit die Sicht auf das Komplex-Objekt. (Beachte: die anderen Eigenschaften aus Abb. 5 beziehen sich direkt auf das Komplex-Objekt und nicht wie hier auf die zugehörige interne Struktur). Aus anwendungsspezifischen Gründen - etwa um Wege im Netz erkennen und bearbeiten zu können - muß der Aufbau von Netzobjekten aus Kantenobjekten erkennbar sein. Hingegen müssen für die Flächenobjekte jeweils nur deren Berandungen sichtbar sein. D.h., für den Flächenobjekttyp bzw. zur Konstruktion der zugehörigen Berandung ist es egal, ob dieser direkt aus dem Knotenobjekttyp oder indirekt mit Hilfe des Kantenobjekttyps modelliert wird. Deshalb sind in Tabelle 1b auch beide Möglichkeiten aufgezeigt. Dabei besagt das LIST-Konstrukt im Gegensatz zum SET-Konstrukt, daß die zugehörige (Objekt-)Menge geordnet ist. Sichtbarkeit der internen Objektstruktur bedeutet, daß die einzelnen Komponenten der internen Struktur zu sehen und damit prinzipiell verarbeitbar sind. Die Möglichkeit, auf diese Interna zuzugreifen und sie zu manipulieren, hat direkte Auswirkungen auf die mit dem Komplex-Objekt assoziierte Operationsmenge und auf die Maßnahmen zur (strukturellen) Integritätserhaltung (siehe Kapitelende und nächstes Kapitel).

Unabhängig von den Struktureigenschaften können die Isolationseigenschaften betrachtet werden, die eine wichtige Erweiterung des intuitiven Komplex-Objekt-Begriffs darstellen. Hierunter fallen Aussagen über die Zugriffsmöglichkeiten auf Komplex-Objekte und über gemeinsam benutzbare - "shared" - (Komponenten-)Objekte. Um diese Eigenschaften spezifizieren zu können, wird eine "Existenz"-Klausel in der Komplex-Objekt-Definition benötigt. Mit Hilfe dieser Klausel wird der Gültigkeitsbereich des Objekttyps festgelegt. Die Existenzabhängigkeit bedeutet dann, daß der betreffende Objekttyp nur als Komponentenobjekttyp verwendet werden darf. Diese

Klasse von Objekttypen bezeichnet man deshalb auch als abhängig (dependent), lokal oder auch nicht-eigenständig. Eine Zusatzoption erlaubt/verbietet dann zusätzlich eine Mehrfachbenutzung der Komponentenobjekte dieses Typs innerhalb der übergeordneten Objekte des Gesamtobjekttyps.
Damit kann man z.B. die gemeinsamen Kantenobjekte benachbarter Flächenobjekte im Komponentenobjekttyp KANTE als nicht-eigenständig (also vom Gesamtobjekttyp FLAECHE existenzabhängig), aber mehrfachbenutzbar definieren und somit zu einer expliziten nichtredundanten Darstellung gelangen. In diesem Fall weiß das zugrundeliegende System über das evtl. Vorkommen von gemeinsamen Kantenobjekten Bescheid. Dadurch, daß der Kantenobjekttyp auch in der internen Objektstruktur des Netzobjekttyps vorkommt (siehe Tabelle 1b), hat das System zusätzliche Kenntnis über gemeinsam verwendete Komponentenobjekttypen in unterschiedlichen Gesamtobjekttypen. Allerdings sind die Kantenobjekte innerhalb von Netzobjekten (für das System) verschieden von den Kantenobjekten der Flächenobjekte. Dieses zusätzliche Wissen des Systems über den Aufbau der Komplex-Objekte kann/sollte zur systemgarantierten (strukturellen) Integritätserhaltung ausgenutzt werden (siehe Kapitelende und nächstes Kapitel).
Die Eigenständigkeit eines Objekttyps definiert einen globalen Gültigkeitsbereich und ermöglicht eine Mehrfachbenutzung der zugehörigen Objekte sowohl innerhalb eines Gesamtobjekttyps als auch zwischen unterschiedlichen Gesamtobjekttypen. Kommt ein globaler Objekttyp mehrfach als Komponentenobjekttyp in verschiedenen Gesamtobjekttypen vor, so sind etwaige gemeinsame Komponentenobjekte dem System ebenfalls bekannt. D.h., es sind dann analog oben auch Integritätsgarantien vom System erbringbar. Unter der Eigenständigkeit eines (Komplex-)Objekttyps wird stets vorausgesetzt, daß die zugehörigen (Komplex-)Objekte mittels eines eindeutigen Bezeichners (u.U. ein Surrogat) als Einheit ansprechbar sind.

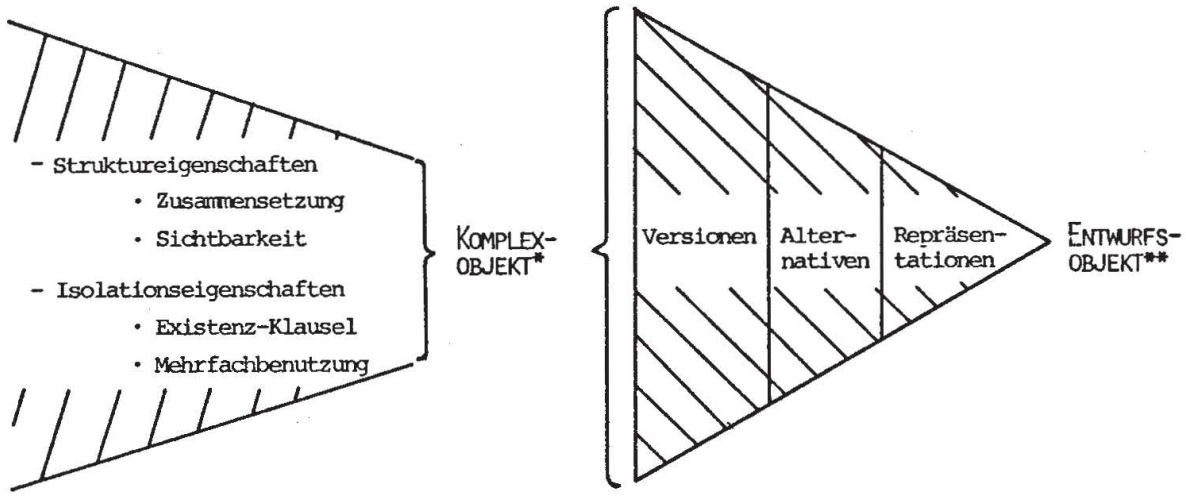
Zwischen der Existenz-Klausel und der oben angesprochenen Zugriffs- und Strukturüberschneidung besteht nun folgender Zusammenhang: Die Eigenständigkeit eines (Komplex-)Objekttyps gestattet die Möglichkeit, sowohl explizit über diesen Objektbezeichner als auch implizit über die evtl. existierende übergeordnete Struktur auf die Objekte dieses Typs zuzugreifen. Umgekehrt erlaubt die Nicht-Eigenständigkeit eines (Komplex-)Objekttyps den Zugriff auf die zugehörigen Objekte ausschließlich über die übergeordnete Struktur. Aus diesem Grunde ist die Nicht-Eigenständigkeit nur bei Komponentenobjekttypen spezifizierbar. Eine Besonderheit für Gesamtobjekttypen bzw. für die Gesamtobjekte, die zudem direkte Auswirkungen auf die Integritätsmaßnahmen hat, stellt der Seitenzugriff auf die Objekte der Komponentenobjekttypen dar. Darunter versteht man, daß mehrere Zugriffsmöglichkeiten - entweder indirekt über interne Objektstrukturen oder direkt über explizite Referenzen aufgrund der Eigenständigkeit des Objekttyps - zu einem Komponentenobjekttyp existieren. Bezogen auf das Beispiel in Tabelle 1 bedeutet dies, daß die Flächenobjekte einmal über die zugehörige Partition zugreifbar und manipulierbar sind und zusätzlich noch direkt. Analog oben sollten auch hier vom System Kontrollmechanismen für einen konsistenten Zugriff angeboten werden (siehe unten).

In Beispiel 2 und 3 aus dem vorigen Kapitel kam der Einfluß des anwendungsabhängigen Entwurfsaspekts auf die Komplex-Objekt-Begriffsbildung deutlich zum Vorschein. Die das Komplex-Objekt ebenfalls charakterisierenden Entwurfsinforma-

tionsstrukturen (/DK84/, /Eb83/, /Ka83/, /Ne83/) sind:

- Repräsentationen als gewählte Formen der Darstellung des Entwurfsobjekts. Man unterscheidet dabei die Repräsentationsform (im Beispiel 2 SPEZIFIKATION→SCHALTKREISDIAGRAMM→LAYOUT), die das gleiche Entwurfsobjekt auf verschiedenen hierarchisch angeordneten Ebenen jeweils vollständig beschreibt, von der Ergänzungsform (im Beispiel 2 SPEZIFIKATION→TEXT und SPEZIFIKATION→SCHALTKREISDIAGRAMM; im Beispiel 3 PRODUKTSTRUKTURMODELL(PSM)→PHYSIKALISCHES_MODELL, PSM→TECHNOLOGISCHES_MODELL, PSM→GEOMETRISCHES_MODELL), die jeweils komplementäre, sich gegenseitig ergänzende Darstellungen von Teilaspekten des Entwurfsobjekts beschreibt.
- Für eine Repräsentation eines Entwurfsobjekts können gleichzeitig mehrere, a priori voneinander unabhängige Alternativen existieren, die (hoffentlich) eine gewisse Kontinuität in Richtung auf eine bessere/optimale Lösung einer Teilaufgabe des Entwurfs darstellen.
- Die Versionen beschreiben die zeitliche Entwicklung der zugehörigen Alternative. Die aktuelle/gültige Version zeigt den momentanen Ausgabe- oder Bearbeitungsstand (design state) an. Meistens ersetzt die "neue" Version die vorherige.
- Die Konfiguration beinhaltet eine Zusammenfassung der Daten (Alternativen und Versionen), die die favorisierte Lösung des zu entwickelnden Entwurfsobjekts darstellt.

Von der u.U. komplexen Objektstruktur - im Sinne von Abb. 5 - des Repräsentations- bzw. Versionsobjekts kann abstrahiert werden (siehe Abb. 3 zu Beispiel 2 bzw. Abb. 4 zu Beispiel 3 sowie /DK84/ und /Ne83/). D.h., alle Entwurfsinformationsstrukturen stellen neue Eigenschaften der Komplex-Objekte dar und sind unabhängig von den zuvor eingeführten Struktur- und Isolationseigenschaften.



* gemäß Abbildung 5

** gemäß Abbildung 2

Abbildung 6: Zusammenstellung der wichtigsten Komplex-Objekt-Eigenschaften

In Abbildung 6 sind die wichtigsten Komplex-Objekt-Charakteristiken nochmals zusammengefaßt. (Falls im folgenden von Komplex-Objekten die Rede ist, so werden darunter grundsätzlich alle Eigenschaften aus Abbildung 6 subsumiert). Die dort aufgeführten, jeweils voneinander unabhängigen Eigenschaften lassen nun die in Kapitel 2 geforderte scharfe Begriffsbildung zu. Ein einfacher Vorschlag dafür ist in Abbildung 5 enthalten: Die atomaren Objekte werden NORMALobjekte und die komplexen KOMPLEXobjekte genannt, während die noch zusätzlich mit "nicht-sichtbar" charakterisierten Objekte als PRIMITIVobjekte bezeichnet werden. Wendet man die soeben angegebenen Begriffsdefinition zusammen mit den oben eingeführten Charakteristiken auf die in Kapitel 2 Beispiel 1 beschriebenen geometrischen Daten an, so kann, wie in Tabelle 1 aufgezeigt, eine semantisch hochstehende und genaue Datenstrukturdefinition gemacht werden.

Versucht man die oben eingeführten und genau beschriebenen Komplex-Objekt-Charakteristiken vom semantischen Standpunkt aus zu analysieren und zu begründen, so gelangt man zu folgenden Aussagen: Die Zusammensetzungseigenschaft liefert bzw. modelliert ausschließlich Information bezogen auf die interne Objektstruktur, d.h. reine "Aufbaudaten". Durch die Hinzunahme der Sichtbarkeitseigenschaft hat man jetzt die Möglichkeit, "Primitiv"-Objekte zu bilden. Damit kann eine Prioritätsverschiebung zugunsten des Gesamtobjekts und eine stärkere Zusammengehörigkeit der Komponentenobjekte erzielt werden. Mit Hilfe der Isolationseigenschaften besteht nun zusätzlich noch die Möglichkeit, zwischen reinen "Aufbau-Komponentenobjekten" und wichtigen mehr informationstragenden (Komponenten-)Objekten zu unterscheiden. Erstere dienen ausschließlich zur Bildung des Gesamtobjekts, während letztere oft wichtige Direktzugriffsobjekte bzw. mehrfachbenutzbare Komponentenobjekte bezeichnen. Diese Aussagen - natürlich in detaillierterer Form - lassen sich relativ einfach zu allgemeineren Schemaentwurfsregeln für das zugrunde gelegte Komplex-Objekt-Datenmodell erweitern.

Der hier eingeführte Komplex-Objekt-Begriff hat, wie oben schon mehrfach erwähnt, direkte Auswirkungen sowohl operationaler als auch konsistenzbezogener Art: Das Komplex-Objekt ist nun gleichermaßen Verarbeitungseinheit wie auch Konsistenzeinheit. Mittels der hier betrachteten Charakteristiken werden dem System genauere Angaben über die von ihm zu verwaltenden (Komplex-)Objekte zur Verfügung gestellt. Dieses zusätzliche Wissen sollte dann vom System dazu benutzt werden, um zum einen erweiterte Integritätsgarantien und zum anderen konsistente Operationen (siehe nächstes Kapitel) anzubieten.

Die in diesem Kapitel herausgearbeiteten Eigenschaften von Komplex-Objekten stellen gleichzeitig Anforderungen an geeignete Datenmodelle dar. Aus diesem Grund befaßt sich das nachfolgende Kapitel mit dem momentanen Stand existierender Modellierungs- und Realisierungskonzepte für Komplex-Objekte.

4. Datenmodelle und Realisierungskonzepte für Komplex-Objekte

In diesem Kapitel wird die Untersuchung bestehender bzw. geplanter Ansätze, Konzepte und Methoden zur Lösung der Komplex-Objekt-Problematik - das heißt der anwendungsgerechten Modellierung und Verwaltung der benutzernahen Objekte -, wie sie hier und beispielsweise auch in /GP83/, /Mi84/ und /SP82/ beschrieben wurde, durchgeführt. Zuvor wird aber eine möglichst optimale Wunschlösung skizziert, die als Referenzsystem herangezogen werden kann.

Dieser Lösungsvorschlag gliedert sich natürlicherweise in zwei Teile auf: Der erste Teil besteht aus einem anwendungsbezogenen Datenmodell, welches die Natürlichkeit und Einfachheit in der Darstellung sowie entsprechend höhere Operationen anbieten und die Formulierbarkeit aller Integritätsbedingungen gewährleisten sollte. Der zweite Teil entspricht einer möglichst optimalen Realisierung dieses Datenmodells durch das zugrundeliegende DBS und umfaßt deshalb hauptsächlich Architektur- und Implementierungsaspekte. Zur tieferen Behandlung dieser Thematik kann auf /HR84/ verwiesen werden. Im folgenden werden daher die Datenmodellaspekte schwerpunktmäßig und etwas ausführlicher betrachtet.

Ein Datenmodell (DM), welches eine vollständige, natürliche und einfache Modellierung der in der Anwendung vorkommenden Objekte ermöglichen soll, muß die in Kapitel 3 erarbeiteten Charakteristiken eines Komplex-Objekts (siehe Abb. 6) zu beschreiben erlauben. Damit ist gleichzeitig ein Großteil der objektbezogenen Integritätsbedingungen schon implizit in den modellinhärenten Konsistenzbedingungen enthalten und kann vom zugrundeliegenden DBS automatisch überprüft werden. Die strukturelle Integrität umfaßt größtenteils Bedingungen an die internen Objektstrukturen, wie z.B. die Gesamtobjekt-Komponentenobjekt-Konsistenz, und wird daher als lokale Konsistenz bezeichnet. Hingegen dienen die Abhängigkeiten auf der Entwurfsebene zur Konsistenzerhaltung zwischen Versionen, Alternativen und Repräsentationen und werden daher oft globale Konsistenz genannt. Die Regeln zur Konsistenzwahrung definieren ein sog. Konsistenzmodell (siehe /Ne83/), welches untrennbar zum anwendungsbezogenen Datenmodell gehört. Da die nun definierbaren Komplex-Objekte auch Einheit der Verarbeitung sind, gehören sowohl Operatoren zur Verwaltung der internen, lokalen Struktur als auch Operatoren für die globale Struktur zum Datenmodell.

Operatoren zum Verwalten der globalen Struktur, d.h. der Versionen, Alternativen und Repräsentationen der Entwurfsebene, sind u.a. die folgenden:

CREATE (design_object / representation / alternative / version)
CONTINUE, RESET, RELEASE (version)
SELECT (representation / alternative / version)
DEFINE (configuration).

Alle Operationen auf der Entwurfsebene beziehen sich auf den vom DBS verwalteten Abhängigkeitsgraphen zwischen den Entwurfsinformationsstrukturen Repräsentation, Alternative und Version. Die CREATE-Operation baut entweder einen neuen Graphen für ein zu kreierendes Entwurfsobjekt auf oder führt einen schon bestehenden Graphen fort. CONTINUE, RESET und RELEASE führen eine spezifizierte Version fort oder setzen diese zurück bzw. geben sie frei. Die SELECT-Operation aktiviert eine

bestimmte Repräsentation, Alternative und/oder Version gemäß den Regeln des Konsistenzmodells (s.o.). Schließlich wird durch die DEFINE-Operation eine neue Konfiguration definiert. D.h., der Abhängigkeitsgraph wird festgeschrieben und die ausgewählten Alternativen und Versionen markiert.

Die lokale Struktur, d.h. die interne Objektstruktur, wird beispielsweise mittels nachstehender Operationen verarbeitet:

INSERT, DELETE, MODIFY, FETCH (complex_object / component_object)
SELECT (child or component_object / parent_object)
COPY, REPLACE (complex_object / component_object).

Das besondere an diesen Operationen im Vergleich zu den (gleichnamigen) Operationen in herkömmlichen Datenmodellen sind die Objekte, auf die bezug genommen wird sowie die dem Datenmodell innewohnenden bzw. vom System garantierten Integritätszusagen. Es können sowohl Komplex-Objekte als auch Komponentenobjekte konsistent verarbeitet werden. Für unsere Geographie-Anwendung aus Kapitel 2 bzw. Tabelle 1 bedeutet dies folgendes (die hier gewählte Syntax ist stark vereinfacht und daher in großem Maße selbsterklärend):

- Holen eines vollständigen Komplex-Objekts inklusive der gesamten dazugehörigen internen Objektstruktur

```
FETCH complex_object PARTITION inclusive
WHERE PAR_NUMMER = '999'
```

- Holen eines Komponentenobjekts innerhalb eines spezifizierten Gesamtobjekts

```
FETCH component_object FLAECHE
FROM complex_object PARTITION
WHERE PAR_NUMER = '999' AND ...
```

Zusätzlich sollte das Datenmodell noch die Möglichkeit zur Definition von speziellen, anwendungsbezogenen Operationen auf der Komplex-Objekt-Ebene (z.B. teile/vereinige Parzelle, Test auf Schnitt zwischen KANTEN- und KANTEN- bzw. FLAECHEN-Objekt) sowie von allgemeinen, datenmodell-fernen Operationen, wie z.B. numerische oder geometrische Operationen (Abstands- und Winkelberechnungen), vorsehen. In letzter Konsequenz zielen diese Forderungen darauf ab, operationale Schnittstellen bzw. entsprechende Definitionsmöglichkeiten im Stile abstrakter Datentypen (ADT) anzubieten (siehe auch /GP83/ und /Lü83/).
Dann könnte die Beispieloperation "Test, ob sich zwei Flächen überschneiden bzw. berühren" wie folgt vereinfacht werden:

- Test, ob beide Flächen in der gleichen Ebene liegen.
  Hierzu sind nur die direkten Attribute (speziell die Flächennormalen) der betreffenden Flächenobjekte zu betrachten.
- Falls ja: Holen der jeweiligen vollständigen Komponentenobjekte BERANDUNG
  Gegenseitiges Verschneiden der Kanten der einen Berandung mit jeder der anderen.
- Falls Schnittmenge nicht leer, dann Überschneidung oder Berührung.

(Eine detailliertere Behandlung der hier nur punktuell skizzierten Operationen ist zweifelsohne notwendig, würde aber den Rahmen dieser Arbeit sprengen und zudem eine thematische Schwerpunktverlagerung bewirken).

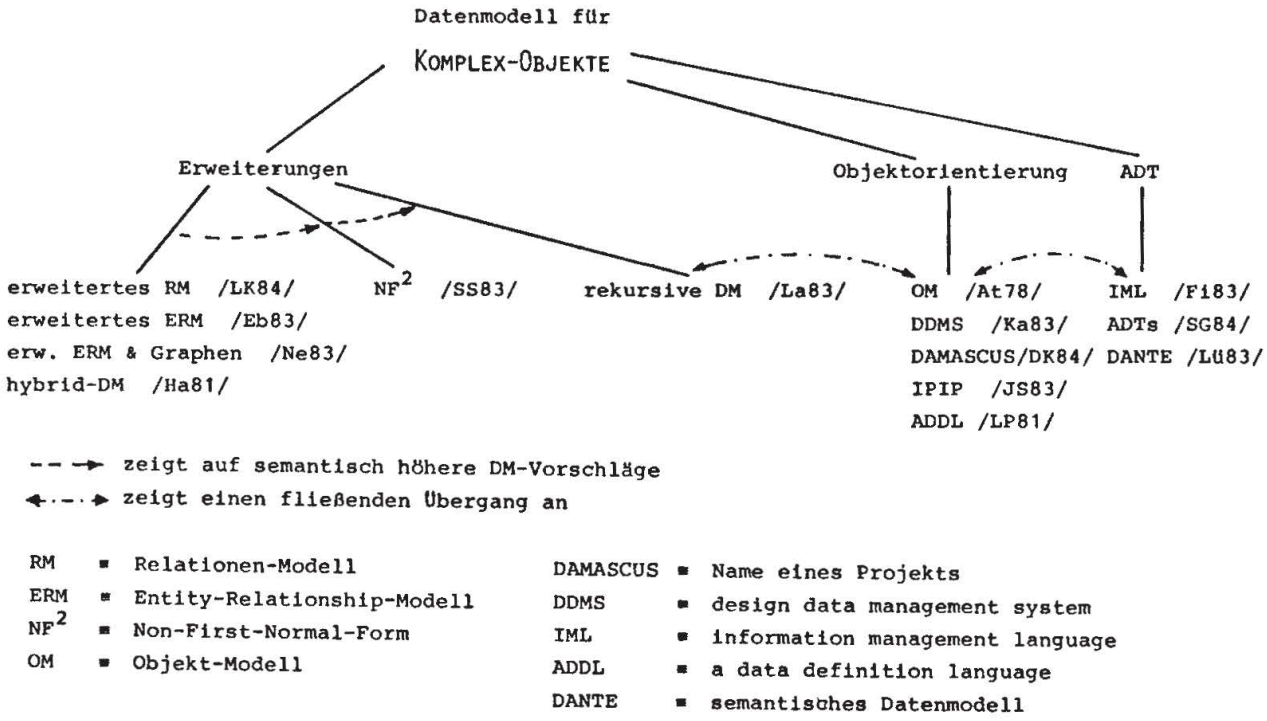


Abbildung 7: Klassifikation von Datenmodellvorschlägen für Komplex-Objekte

Das zuvor skizzierte Datanmodell zusammen mit seinen erweiterten Integritätsgarantien sowie Modellierungs- und Operationsmöglichkeiten repräsentiert unsere favorisierte Lösung und dient im folgenden als Referenzsystem.

Abbildung 7 zeigt eine einfache Klassifikation der hier aufgezeigten Datenmodellvorschläge für Komplex-Objekte. Der Übergang zwischen den einzelnen Klassifikationspunkten ist fließend. Dies bedeutet insbesondere, daß die Konzepte basierend auf abstrakten Datentypen, Objektorientierung und semantisch hohen DM-Erweiterungen ziemlich gleichmächtig sind, jedoch unterschiedliche Schwerpunkte besitzen.

Ein fundierter Vergleich der DM-Vorschläge aus Abb. 7 mit unserem Referenzmodell würde den engen Rahmen dieser Arbeit bei weitem sprengen. Zudem erscheint eine Beurteilung sowohl kozeptionell als auch von der Implementierungsseite her sehr problematisch, da die einzelnen DM-Konzepte oft unterschiedlichen Systemebenen zuzuordnen sind oder gar manche sich gerade im Definitionsstadium befinden bzw. nie darüberhinausgekommen sind. Deshalb werden im folgenden die unserer Meinung nach wichtigsten Aspekte der interessantesten DM-Konzepte kurz beschrieben, die dennoch einige wichtige Aussagen abzuleiten erlauben.

Der NF^2-Vorschlag (/PS84/ und /SS83/) umfaßt ein elegantes, um den Komplex-Objekt-Aspekt erweitertes, relationales Datenmodell und eine effiziente Implementierung desselben. Das NF^2-Relationenmodell ist charakterisiert durch die Zulassung nichtatomarer Attributwerte, d.h. durch die Aufgabe der 1. Normalform. Damit sind relationenwertige Attribute zulässig, was einer expliziten Darstellung von hierarchisch gegliederten (Komplex-)Objekten gleichkommt. Die Modellierung nichthierarchischer Beziehungen ist jedoch nur mit entsprechender Redundanz möglich. Zur

Verarbeitung dieser NF^2-Hierarchien wird eine erweiterte, rekursive Relationenalgebra bereitgestellt, die sowohl den Zugriff auf die übergeordnete Relation (im Sinne des Gesamtobjektzugriffs) als auch auf die relationenwertigen Attribute (im Sinne des Komponentenobjektzugriffs) erlaubt.

DANTE (/Lü83/) bezeichnet ein semantisches Datenmodell und zeigt einen operationalen Ansatz zur Modellierung/Verarbeitung technischer Objekte aus dem Anwendungsbereich Konstruktion. Zusammen mit der Definition von Komplex-Objekten soll die Spezifikation anwendungsabhängiger Operatoren zur Manipulation der Objekte und zur Selektion von Daten über die Objekte ermöglicht werden. DANTE wird realisiert als Programmiersprachenerweiterung von MODULA-2 (siehe /Wi82/) und kommt der geforderten ADT-ähnlichen DBS-Schnittstelle ziemlich nahe.

Die in /La83/ beschriebenen rekursiven Datenmodelle ermöglichen die Definition und Verarbeitung von Komplex-Objekten, die nicht notwendig gleiche (Objekt-)Strukturen, wie bei den klassischen Datenmodellen, sondern nur noch gleichartige Strukturierungsprinzipien haben müssen. Sie bauen auf rekursiven Typdefinitionen mit jeweils verschiedenen elementaren Strukturgeneratoren und Komponentenselektoren auf. Der Ansatz integriert Erfahrungen aus den Bereichen Programmiersprachen und (klassische) Datenmodelle.

Manche Vorschläge sind nur für genau einen Aspekt des Komplex-Objekts geeignet, d.h. entweder für die Objekt- oder für die Entwurfsstrukturen.

Das in /LK84/ beschriebene erweiterte Relationenmodell und dessen Implementierung im relationalen DBS SYSTEM R (/As76/) zeigt nur für die Objektstrukturen eine sehr gute Bewertung. Die anderen Aspekte werden dagegen nur rudimentär unterstützt. Der Ansatz stellt die Objektstrukturen, d.h. die Struktur- und Isolationseigenschaften, sowohl logisch auf semantischer Ebene als auch physisch auf Implementierungsniveau vor. Die Modellierungskonzepte erlauben ziemlich alle in Kap. 3 aufgeführten Nuancen von Komplex-Objekten darzustellen. Auch werden dort Zugriffspfadstrukturen zu den Komplex-Objekten angegeben sowie Operationen auf Komplex-Objekten, die eine höhere Verarbeitungsschnittstelle bereitstellen.

Ein anderer Ansatz, der speziell für die Entwurfsstrukturen geeignet erscheint, wird in /Ne83/ vorgestellt. Dieser basiert auf einem Zweiebenenmodell. Die Entwurfsebene enthält das sog. Entwurfsschema in Form von Abhängigkeiten zwischen verschiedenen Repräsentationen. Die Entwurfsobjekte sind die Ausprägungen dieses Schemas und werden als Graphstruktur über den Versionen und Alternativen der Repräsentationen dargestellt. Die Repräsentationsebene beschreibt die Struktur der im Schema definierten Repräsentationen, die u.U. vom Typ Komplex-Objekt sein kann. Die Darstellungsmöglichkeiten dieser zweiten Ebene erscheinen allerdings gegenüber anderen Vorschlägen ziemlich dürftig (vgl. etwa /SS83/, /Lü83/). Die Abhängigkeiten zwischen Versionen, Alternativen und Repräsentationen auf der Entwurfsebene dienen zusammen mit den von der Entwurfsmethodologie abhängigen Integritätsregeln der Konsistenzerhaltung. Die Verwendung der allgemeinen Graphstruktur erlaubt eine elegante Realisierung und Einhaltung des gültigen Konsistenzmodells.

Die hier grob analysierten Konzepte zeigen die prinzipiell möglichen Ansatzpunkte zur Lösung der anfangs aufgestellten Komplex-Objekt-Problematik. Die Palette der Vorschläge reicht von erweiterten bzw. neuen und semantisch hohen Datenmodellen bis zu Implementierungs- und Optimierungsvorschlägen. Die weitestgehend voneinander getrennten Ansätze müßten integriert werden. Erst dann kann, wie oben schon bemerkt, die auf DM-Ebene verfügbare (semantische) Information vollständig ausgenutzt werden. Sinnvolle Verknüpfungsmöglichkeiten bestehen zum einen zwischen den Relationenmodell-Erweiterungen aus /LK84/ und höheren DM-Vorschlägen, wie z.B. in /La83/ oder in /LP81/. Zum anderen kann man auch die Kombination der Ansätze aus /LK84/ und /Ne83/ mit den o.g. höheren DM-Vorschlägen noch zusätzlich verknüpfen. Diese Vorschläge sind für eine Realisierung auf herkömmlichen DBS geeignet. Damit kann relativ schnell und mit vertretbarem Aufwand ein Prototyp zum "Kennenlernen" bzw. zum "Sammeln von Erfahrungen" entwickelt werden. Das Arbeits- und Leistungsverhalten sowie etwaige Engpässe des Prototyps sind dann analysierbar. In Form von Verbesserungshinweisen und Entwicklungsrichtlinien sind diese Informationen dann nutzbar für die Entwicklung von neuen und den Anforderungen angepaßten Non-Standard-DBS.

5. Zusammenfassung

Die Motivation zur Behandlung dieser Thematik lag in der Notwendigkeit der anwendungsgerechten Modellierung und Verarbeitung der komplexen (Anwendungs)-Objekte in Datenbanksystemen, die für den Einsatz in Non-Standard-Anwendungen vorgesehen sind.

Die vielfältigen Nuancen des Komplex-Objekt-Begriffs verursachen eine extreme Begriffsüberladung und Begriffsvielfalt, die eine wissenschaftliche Diskussion der Komplex-Objekt-Verwaltung sehr erschweren. Durch die Erarbeitung der inhärenten Komplex-Objekt-Charakteristiken wird eine Begriffsklärung erreicht. Unter Verwendung dieses formalisierten Komplex-Objekt-Begriffs wird ein Komplex-Objekt-Datenmodell inklusive des zugrundeliegenden Datenbanksystems stichpunktartig vorgestellt. Dieses Referenzsystem kann als Bewertungsmaßstab für bislang existierende bzw. geplante Realisierungskonzepte dienen.

Die hier vorgestellte Formalisierung des Komplex-Objekt-Begriffs muß weiter verfeinert werden. Ebenso sind die vielfältigen Abhängigkeiten zwischen Komplex-Objekt-Datenmodell und Realisierung desselben im darunterliegenden Datenbanksystem detaillierter zu untersuchen.

Ich danke Herrn Prof. Dr. T. Härder für die Anregung, mich mit diesem Thema zu befassen sowie für seine hilfreichen Anmerkungen während der Entstehungsphase dieser Arbeit. Bei meiner Kollegin Frau Andrea Sikeler und bei meinem Kollegen Herrn Klaus Küspert möchte ich mich für das sorgfältige Korrekturlesen des Manuskripts bedanken sowie bei den Referees für die hilfreichen Anmerkungen und Verbesserungsvorschläge.

Literaturverzeichnis

As76 Astrahan, M. M., et. al.: SYSTEM R: Relational Approach to Database Management, in ACM Trans. on Database Systems Vol. 1, No. 2, 1976, pp. 97 - 137

At78 Athay, R. J.: Object Models for Computer Aided Design: An Overview, in Computer Graphics, Vol. 12, No. 3, 1978, pp. 239 - 244

DK84 Dittrich, K., Kotz, A.: DAMASCUS - Projektvorstellung, Vortrag beim Treffen der "Regionalgruppe Südwest", Heidelberg, 1984

DP84 Dadam, P., Pistor, P.: AIM-Projektüberblick, Vortrag beim Treffen der "Regionalgruppe Südwest", Heidelberg, 1984

Ea80 Eastman, C. M.: System Facilities for CAD-Databases, in: Proc. of 17th Design Automation Conf., Minneapolis, 1980, pp. 50 - 56

Eb83 Eberlein, W.: Architektur technischer Datenbanken für Integrierte Ingenieursysteme, Dissertation, Technische Fakultät der Universität Erlangen-Nürnberg, 1983

EW81 Eberlein, W., Wedekind, H.: A Methodology for Embedding Design Databases into Integrated Engineering Systems, in: Proc. of the IFIP Conf. on CAD Data Bases, North-Holland Publ. Co., 1981

Fi83 Fischer, W. E.: Datenbanken für CAD-Arbeitsplätze, Informatik-Fachberichte Nr. 70, Springer-Verlag, 1983

Fr83 Frank, A.: Datenstrukturen für Landinformationssysteme - Semantische, topologische und räumliche Beziehungen in Daten der Geo-Wissenschaften, Dissertation, ETH Zürich, 1983

GP83 Gründig, L., Pistor, P.: Land-Informations-Systeme und ihre Anforderungen an Datenbank-Schnittstellen, in: Informatik Fachberichte 72, Sprachen für Datenbanken, Fachgespräch auf der 13. GI-Jahrestagung in Hamburg, 1983, S. 61 - 75

Ha81 Haynie, M. N.: The Relational/Network Hybrid Data Model for Design Automation Dababases, in Proc. of the 18th IEEE Design Autom. Conf., 1981, pp. 646 - 652

HR85 Härder, T., Reuter, A.: Architektur-Konzepte von Datenbanksystemen für nicht-kommerzielle Anwendungen, erscheint in diesem Tagungsband

JS83 Johnson, H. R., et. al.: A DBMS Facility for Handling Structured Engineering Entities, in Proc. of the Engineering Design Application at the Data Base Week, 1983, pp. 3 - 11

Ka83 Katz, R.: Managing the Chip Design Database, Computer Sciences Technical Rep. No. 506, University of Wisconsin-Madison; auch in IEEE Computer, Dec. 1983

La83 Lamersdorf, W.: Rekursive Datenmodelle, in: Informatik Fachberichte 72, Sprachen für Datenbanken, Fachgespräch auf der 13. GI-Jahrestagung in Hamburg, 1983, S. 148 - 168

LK84 Lorie, R., Kim, W., et. al.: Supporting Complex Objects in a Relational System for Engineering Databases, IBM Research Report, IBM Research Laboratory, San Jose California, 1984

Lo81 Lorie, R.A.: Issues in Databases for Design Applications, in: Proc. of the IFIP Conf. on CAD Data Bases, North-Holland Publ. Co., 1981

Lo83 Lohman, G.: Remotely-sensed Geophysical-Databases: Experience and Implications for Generalized DBMS, IBM Res. Rep., No. RJ3794, 1983

LP81 Lacroix, M., Pirotte, A.: Data Structures for CAD Object Description, in Proc. of the 18th IEEE Design Automation Conf., 1981, pp. 653 - 659

Lu83 Lum, V.: Advanced Information Management (AIM) Projektüberblick, Vortrag beim "Non-Standard DB Workshop", Heidelberg, 1983

Lü83 Lüke, B.: DANTE - Ein semantisches Datenmodell für Anwendungen aus dem Konstruktionsbereich, Interner Bericht, Universität Karlsruhe, 1983

Mi84 Mitschang, B.: Überlegungen zur Architektur von Datenbanksystemen für Ingenieursanwendungen, im Tagungsband der 14. GI-Jahrestagung in Braunschweig, 1984, S. 318 - 334

Ne83 Neumann, Th.: On representing the design information in a common database, in Proc. of the Engineering Design Applications at the Data Base Week, 1983, pp. 81 - 87

PS84 Paul, H.-B., Schek, H.-J., Scholl, M., Weikum, G.: Überlegungen zur Architektur eines "Non-Standard"-Datenbankkernsystems, Arbeitsbericht DVSI1984-A2, TH Darmstadt

Si80 Sidle, T. W.: Weaknesses of Commercial Data Base Management Systems in Engineering Application, in: Proc. of 17th Design Automation Conf., Minneapolis, 1980, pp. 57 - 61

SG84 Stonebraker, M., Guttman, A.: Using a Relational Database Management System for Computer Aided Design Data - An Update, in IEEE Database Engineering, Vol. 7, No. 2, 1984. pp. 56 - 60

SP82 Schek, H.-J., Pistor, P.: Data Structures for an Integrated Data Base Management and Information Retrieval System, in: Proceedings of the 8th VLDB-Conf., 1982

SS83 Schek, H.-J., Scholl, M.: Die NF^2-Relationenalgebra zur einheitlichen Manipulation externer, konzeptieller und interner Datenstrukturen, in: Informatik Fachberichte 72, Sprachen für Datenbanken, Fachgespräch auf der 13. GI-Jahrestagung in Hamburg, 1983, S. 111 - 133

Wi82 Wirth, N.: Programming in MODULA-2, Springer Verlag Berlin, Heidelberg, New York, 1982

Diese Arbeit entstand im Rahmen eines Projektes innerhalb des von der Deutschen Forschungsgemeinschaft geförderten Sonderforschungsbereichs 124.