in: Tagungsband der GI-Fachtagung ‘Datenbanken in Büro, Technik und Wissenschaft’ (BTW’95), Informatik aktuell, Dresden, März 1995, pp. 348-357.

Ein objektorientierter Ansatz zur Restrukturierung der betrieblichen Informationsverarbeitung

Günter Sauter[1], Joachim Thomas
Universität Kaiserslautern, FB Informatik
67653 Kaiserslautern
e-mail: thomas@informatik.uni-kl.de

Überblick

In vielen Unternehmen wird der aktuelle Datenbestand durch heterogene Datenbank- und Anwendungssysteme verwaltet. Die Restrukturierung der betrieblichen Informationsverarbeitung ist ein Weg, um dieses Altlastenproblem zu bewältigen. Im vorliegenden Aufsatz werden die Ergebnisse eines Projekts dargestellt, das ein Unternehmen der Versicherungsbranche (R+V-Versicherung[2]) in Zusammenarbeit mit der AG Datenverwaltungssysteme der Universität Kaiserslautern durchführte. Anlaß des Projekts war die EG-weite Liberalisierung des Versicherungsmarkts, die eine beträchtliche Diversifikation der am Markt angebotenen Versicherungsprodukte mit sich brachte. Dies verlangt ein hohes Maß an Flexibilität in der Informationsverarbeitung, was durch die bestehenden, heterogenen Anwendungssysteme der R+V-Versicherung nicht mehr gewährleistet war. Im Rahmen des Projekts wurde ein repräsentativer Ausschnitt der Informationsverarbeitung mit Methoden der objektorientierten Analyse strukturiert und mit Hilfe des Wissensbankverwaltungssystems KRISYS beispielhaft modelliert. Abschluß des Projekts war die Validierung der Ergebnisse hinsichtlich ihrer Adäquatheit für die Erfordernisse der R+V-Versicherung und ihrer Realisierbarkeit mittels kommerziell verfügbarer DBMS.

Stichworte: Altlastenproblem, Objektorientierte Analyse, Objektorientiertes Design, Reverse Engineering.

1. Einleitung

Die Restrukturierung der betrieblichen Informationsverarbeitung ist eine Aufgabe, die für viele Unternehmen immer mehr an Bedeutung gewinnt. Da der Einsatz von EDV noch bis vor wenigen Jahren mit sehr hohen Kosten verbunden war, gab es in den wenigsten Unternehmen ein umfassendes Konzept zur computergestützten Datenhaltung. Stattdessen wurden für ausgewählte Bereiche Einzelsysteme angeschafft. Der daraus resultierende hohe Grad an Heterogenität in der Datenverwaltung wurde zudem oft durch die Entwicklung der Unternehmen selbst (Expansion, Fusion mit anderen Firmen) bedingt.

Unmittelbare Konsequenz dieser Heterogenität sind hohe Betriebskosten. Sie entstehen zum einen aus der Notwendigkeit, unterschiedliche Systeme getrennt voneinander bedienen und warten zu müssen, zum anderen durch die im allgemeinen nicht zu vermeidende Redundanz der Daten, die zu höherem Speicherplatz- und Verwaltungsaufwand führt. Eine weitere Auswirkung, die unter Umständen weit größere finanzielle Einbußen verursacht, ist die Inflexibilität einer Ansammlung heterogener Anwendungssysteme. Systemübergreifende Zugriffe auf Daten sind im allgemeinen unmöglich, wodurch betriebliche Aufgaben erschwert bzw. verhindert werden. Durch die zunehmende Verfügbarkeit leistungsfähiger und kostengünstiger

---

1. Daimler-Benz AG, Forschungszentrum Ulm, Produktionsinformatik (F3P), Postfach 2360, 89013 Ulm, e-mail: guenter.sauter@dbag.ulm.DaimlerBenz.COM
2. R+V Allgemeine Versicherung, Taunusstr. 1, 65193 Wiesbaden.

EDV-Lösungen verstärkt sich der Druck auf die Unternehmen, dieses *Altlastenproblem (legacy problem)* [BS93, EKPR92] zu bewältigen.

1.1 Alternative Ansätze zur Restrukturierung

Ein Ausweg aus dieser Situation ist die Restrukturierung der betrieblichen Informationsverarbeitung. Dabei lassen sich prinzipiell zwei Ansätze unterscheiden. Der idealtypische Weg ist die vollständige Neuentwicklung von Anwendungssystemen auf Basis eines unternehmensweiten Daten- und Funktionsmodells. Diese Lösung ist jedoch in den wenigsten Fällen praktikabel, zum einen, weil der finanzielle Aufwand sehr hoch ist (Reorganisation des gesamten Datenbestandes, Sperren des gesamten Datenbestandes während der Migrationsphase[3]), zum anderen, weil die zur Realisierung notwendigen personellen Kapazitäten nicht ohne weiteres von den laufend anfallenden EDV-Aufgaben entbunden werden können [EKPR92].

Der alternative Weg zur Restrukturierung der betrieblichen Informationsverarbeitung besteht in der Integration von neuen und alten Systemen, wobei die bestehende Umgebung genutzt wird, um eine schrittweise und für den Benutzer transparente Migration von Anwendungs- und Datenverwaltungssystemen zu erreichen [RS94]. Diese Vorgehensweise wird durch *Föderierte Datenbanksysteme* [SL90] unterstützt. Kennzeichen dieses Architekturansatzes sind die gemeinsame Verwaltung möglicherweise heterogener Datenbanksysteme, die Autonomie integrierter lokaler (oder Alt-) Systeme und die Verteilung von Daten über mehrere Datenbanken der Föderation hinweg.

Ganz gleich, ob die Restrukturierung der betrieblichen Informationsverarbeitung durch Integration von Altsystemen oder durch Neuentwicklung erreicht werden soll, beiden Ansätzen muß ein *einheitliches Daten- und Funktionsmodell* zugrunde liegen, was aus der Analyse der unternehmensweiten Informationsstrukturen und Informationsflüsse gewonnen werden kann. Dieser Vorgang wird oft als *Reverse Engineering* bezeichnet [CC90].

1.2 Die Informationsverarbeitung der R+V-Versicherung

Im vorliegenden Aufsatz werden die Ergebnisse eines Projekts dargestellt, das die R+V-Versicherung in Zusammenarbeit mit der AG Datenverwaltungssysteme der Universität Kaiserslautern durchführte. Anlaß zu diesem Projekt war die EG-weite Liberalisierung des Versicherungsmarkts zum 01.07.1994, die es Versicherungsunternehmen erlaubt, ihre Produkte innerhalb der gesamten EG zu vertreiben. Diese geänderten Rahmenbedingungen führen zu einer beträchtlichen Diversifikation der am Markt angebotenen Versicherungsprodukte. Dies betrifft einerseits standardisierte, einfache Produkte, andererseits werden Produkte benötigt, die individuell auf die Bedürfnisse einzelner Kunden abstimmbar sein müssen. Zusätzlich besteht die Notwendigkeit, auf eine geänderte Nachfrage schnell und kostengünstig reagieren zu können. Zur Wahrung der Konkurrenzfähigkeit müssen neue, zielgruppengerechte Produkte innerhalb sehr kurzer Zeit auf dem Markt angeboten werden können. Geeignete Anwendungssysteme müssen ein hohes Maß an Flexibilität aufweisen, um auf Änderungen des betrieblichen Regelwerks (z.B. die einer Versicherung zugrundeliegenden Tarife oder die von ihr abgedeckten Ereignisse) schnell und unkompliziert reagieren zu können. Diesen Ansprüchen werden die bestehenden Anwendungssysteme der R+V-Versicherung nicht mehr gerecht. Im Rahmen des Projekts sollte daher untersucht werden, inwieweit neuartige Datenmodelle und Modellierungstechniken zur Lösung der Anforderungen beitragen können.

Die Untersuchungen wurden für einen zentralen Bereich der Informationsverarbeitung der R+V-Versicherung, das Teilmodell *Produkt*, durchgeführt. Dieses Teilmodell wurde mit

3. Komplexe Abbildungen zwischen den Schemata von Alt- und Neusystemen, ein hoher Grad an Redundanz, viele unterschiedliche Altsysteme, etc. führen zu einer unerwünscht langen Migrationsphase.

Methoden der objektorientierten Analyse (OOA) strukturiert. Die auf diese Weise ermittelten Informationsstrukturen und -flüsse wurden mit Hilfe von KRISYS [Ma91, DLMT93], einem an der Universität Kaiserslautern entwickelten Wissensbankverwaltungssystem (WBVS), beispielhaft modelliert. Die Vielfältigkeit des von KRISYS angebotenen Wissensmodells erlaubte dabei die direkte Umsetzung der in der OOA gewonnenen Eigenschaften des Teilmodells *Produkt*. Da KRISYS als prototypisches System nicht den Leistungsanforderungen kommerzieller Anwendungen gewachsen ist, kann diese Implementierung nur als ein erster Schritt in Richtung objektorientiertes Design (OOD) angesehen werden[4]. So wurde, ergänzend zur Modellierung mit KRISYS, zum Abschluß des Projekts untersucht, inwieweit das aus der OOA gewonnene Daten- und Funktionsmodell auf ein kommerzielles (objektorientiertes) DBMS abgebildet werden kann.

1.3 Inhaltsübersicht

Das Teilmodell *Produkt* der R+V-Versicherung wird in Abschnitt 2 beschrieben. In Abschnitt 3 werden die grundlegenden Konzepte der objektorientierten Analyse und des objektorientierten Designs nach Coad/Yourdon vorgestellt. Nachdem in Abschnitt 4 das WBVS KRISYS beschrieben wurde, diskutiert Abschnitt die Realisierung des objektorientierten Entwurfs mit KRISYS. Der Aufsatz schließt mit einer zusammenfassenden Bewertung der Ergebnisse des Projekts (Abschnitt 6).

2. Das Teilmodell Produkt

Historisch bedingt existieren bei der R+V-Versicherung heterogene Anwendungssysteme für einzelne Versicherungssparten. Die wesentlichen Nachteile, die sich aus dieser Situation ergeben, sind zum einen ein sehr hoher Grad an Redundanz (dies betrifft sowohl die zu verwaltenden Daten, als auch die zu ihrer Manipulation verfügbaren Dienstprogramme), zum anderen ist es unmöglich, auf einfache Weise spartenübergreifende Zugriffe durchzuführen, z.B., um alle von einer Person abgeschlossenen Versicherungen zu erfragen. Die Restrukturierung der Informationsverarbeitung der R+V-Versicherung muß infolgedessen darauf abzielen, einen einheitlichen Rahmen zu schaffen, der es erlaubt, Informationen und Funktionalitäten weitestgehend wiederzuverwenden und auf diese Weise Redundanz zu vermeiden.

Um die Vorteile und Schwierigkeiten auszuloten, die sich aus solchen Restrukturierungsmaßnahmen ergeben, entschied sich die R+V-Versicherung dafür, ein Pilotprojekt durchzuführen. Zu diesem Zweck mußte ein Teilbereich ausgewählt werden, der für alle Sparten des Unternehmens prototypische Eigenschaften aufweist. Ein solcher Bereich ist das Teilmodell *Produkt*, in dem Produkte (z.B. Versicherungen) und die mit ihnen verknüpften Produktaktionen (z.B. Vertragsabschluß) beschrieben werden. Der Begriff *Produkt* spielt in allen Bereichen des Unternehmens eine wichtige Rolle, so daß einerseits der Aspekt der *Allgemeingültigkeit* zu berücksichtigen ist (durch eine geeignete Spezifikation von Produktschablonen) und andererseits die Eigenschaft der *Wiederverwendbarkeit* anhand von Produkten verschiedener Sparten validiert werden kann. So muß es beispielsweise möglich sein, aus den allgemeinen Vorgaben auch sehr spezielle Produkte entwickeln zu können. Desweiteren bestehen Produkte nicht nur aus einem statischen Teil, sondern besitzen auch *dynamische Eigenschaften*, so daß deklarative und operationale Aspekte gleichermaßen Beachtung finden müssen.

4. Objektorientiertes Design ist nach [FK92] ein Prozeß, im dem die in der Analyse ermittelten Anforderungen in eine spezifische, systembasierte Implementierung übertragen werden, wobei Kosten- und Leistungsziele zu berücksichtigen sind.

2.1 Elementare Anforderungen an das Teilmodell Produkt

Um dem Wunsch nach Flexibilität gerecht zu werden, war ein Baukasten für Versicherungsprodukte zu entwerfen und zu realisieren, mit dessen Hilfe es zunächst einmal möglich sein sollte, Produkte und deren Bestandteile angepaßt und einheitlich zu modellieren. Desweiteren sollte mit dieser Darstellung erreicht werden, daß bestehende Produkte in einfacher Weise modifiziert bzw. neue Produkte flexibel und unter weitestmöglicher Wiederverwendung bereits existierender Produkte /Produktbausteine definiert werden können.

Neben den statischen Eigenschaften von Produkten sollten produktbezogene und produktübergreifende Operationen in den Produktbaukasten integriert werden, so daß die mit Produkten verbundenen Aktionen (z.B. Abschluß/Inanspruchnahme/Kündigung einer Versicherung) auf sinnvolle Weise rechnergestützt ablaufen können.

Eine weitere wesentliche Anforderung an den Produktbaukasten war es, Mechanismen bereitzustellen, um Produkte flexibel zu Produktpaketen kombinieren zu können.

2.2 Genauerer Blick auf das Teilmodell

Grundlage des Projekts war eine Beschreibung der Informationsverarbeitung der R+V-Versicherung, die in einer Art Entity-Relationship-Diagramm vorlag [R+V92]. Es unterteilt sich in statische und dynamische Aspekte. Diese geben einen ersten Überblick über die unternehmensweite Strukturierung der Daten und Abläufe und spezifizieren Anforderungen an ein später zu erstellendes, konkretes Modell. Somit lag eine Grobstrukturierung der Daten und der damit verbundenen Aktionen vor, die innerhalb des Projekts zu verfeinern und zu validieren war.

Das Teilmodell *Produkt*, wie es von der R+V-Versicherung vorgegeben wurde [R+V92, Sa94], wird in Bild 1 grafisch dargestellt. Die Entities auf der linken Hälfte des Diagramms beschreiben einen Produktbaukasten. Das so dargestellte verkaufsfähige Produkt wird durch eine Produktaktion im Rahmen eines Vertragsabschlusses oder einer (versuchten) Inanspruchnahme genutzt. Die Struktur, die zur Aufnahme der dabei anfallenden Informationen dient, soll durch die Entities auf der rechten Seite von Bild 1 realisierbar sein. Es folgt nun eine detailliertere Beschreibung des Diagramms.

Zu einem ***Produkt*** gehören die Eigenschaften des Produkts selbst, sowie Regeln, die bei dessen Verwendung einzuhalten sind. Das Produkt setzt sich aus einem Baukasten bestehend aus Produktkomponenten (***Prodkomp***) zusammen. So besteht z.B. eine ‘Vollkaskoversicherung’ u.a. aus dem Baustein ‘Teilkaskoversicherung’.

Produkte setzen sich (rekursiv) aus Produktkomponenten zusammen. Diese Struktur wird über die Beziehung ***Prodkomp-ZS*** (ZS für Zusammensetzung) dargestellt. Desweiteren können Produktkomponenten oder Produktbestandteile untereinander in Beziehung stehen. So kann beispielsweise das Ereignis ‘Wildunfall’ mit dem Schadensfall ‘Totalschaden’ zusammenhängen. Solche Zuordnungen werden durch die Beziehung ***Prodkomp-ZO*** (ZO für Zuordnung) erfaßt.

Produktaktionen sind Produkten zugeordnet und beschreiben eine Produktnutzung. So gehört zu jedem Produkt die Produktaktion ‘Vertragsabschluß’. Produktaktionen erzeugen neue Informationen, die i.a. von der Struktur des zugrunde liegenden Produkts abhängen. Wie oben beschrieben, ist ein Produkt aus Produktkomponenten aufgebaut; folglich sollte auch für die Ergebnisse von Produktaktionen das Baukastenprinzip verfügbar sein. Zu diesem Zweck dienen die Strukturen ***Produktnutzung*** bzw. ***Prodkomp-Nutzung-ZS***. Durch sie werden beispielsweise die Auswirkungen des Erwerbs oder der (versuchten) Inanspruchnahme eines Produkts beschrieben. Anhand von ***Prodkomp-Nutzung-ZO*** können analog zu den Produktkomponenten die Nutzungskomponenten einander zugeordnet werden. Das Ergebnis des oben erwähnten Vertragsabschlusses wird beispielsweise in der Struktur ***Produktvereinbarung*** festgehalten; die

Struktur ***Produktnachweis*** nimmt alle nach einer Produktvereinbarung aufgetretenen Vorfälle auf.

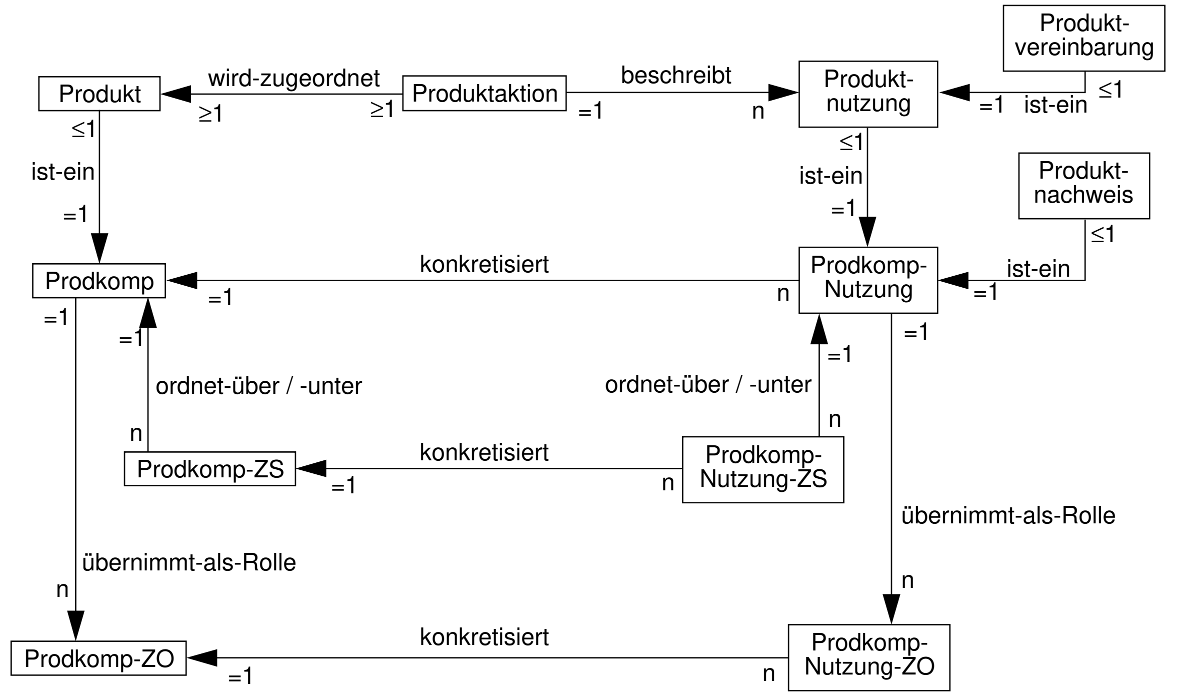


Bild 1: Das Teilmodell Produkt

Dieser Ausschnitt des Unternehmensmodells wurde mittels der objektorientierten Analyse (OOA) verfeinert und mit dem WBVS KRISYS abgebildet. Daher sollen zunächst Konzepte der OOA und OOD kurz vorgestellt werden.

3. Objektorientierte Analyse und Design nach Coad/Yourdon

Im Gegensatz zu klassischen strukturierten Analyse- und Entwurfsmethoden, die auf herkömmliche Programmiersprachen abgestimmt sind, bauen objektorientierte Verfahren auf der objektorientierten Programmierung auf [FK92]. Zentrale Bestandteile des Objektmodells sind [Bo94]:

- die *Abstraktion*, um die wesentlichen Eigenschaften eines Objekts zu erfassen und es klar von anderen Objekten zu unterscheiden,
- die *Kapselung*, um Details eines Objekts vor der Außenwelt zu verbergen, weil sie außerhalb des Objekt nicht relevant sind oder geheimgehalten werden sollen,
- die *Modularität*, die die Zerlegbarkeit eines Problemfelds in zusammenhängende aber dennoch klar abgegrenzte Teilbereiche beschreibt, und
- die *Bildung von Hierarchien* mit Hilfe von Generalisierung und Aggregation.

3.1 Objektorientierte Analyse

Es gibt eine Reihe von Verfahren zur objektorientierten Analyse [St93]. Im folgenden stellen wir den von Coad/Yourdon vorgeschlagene Ansatz vor [CY91a]. Die Autoren haben die Analyse in fünf Schritte unterteilt, welche allerdings mehr als Anhaltspunkte bei der Durchführung denn als feste Reihenfolge zu verstehen sind.

Finden von Klassen und Objekten

Ein *Objekt* im Sinne der Modellwelt ist eine Abstraktion eines Objektes der Problemumgebung, die es erlaubt, alle für das System relevanten Informationen zu halten und auf diese zuzugreifen. Eigenschaften und Verhalten des Objektes werden gekapselt. Eine *Klasse* ist definiert als die Beschreibung eines oder mehrerer Objekte mit gleichen Attributen und Methoden, wozu sich auch ein Beschreibung zur Generierung weiterer zur Klasse konformer Objekte gehören muß.

Durch den Aufbau von Klassen und das Sammeln der Objekte wird der Kontext der Problemumgebung grob abgegrenzt und die Statik analysiert.

Strukturierung

Im zweiten Schritt werden die Klassen und Objekte zusammengefaßt, wobei die Abstraktionskonzepte Generalisierung/Spezialisierung und Aggregation verwendet werden können.

Modularisierung

Ein *Modul* ist ein (möglichst abgeschlossener) Teilbereich eines größeren Problems, der unabhängig modelliert und betrachtet werden kann. Es gibt zwei Möglichkeiten nach Modulen einzuteilen: *top-down* und *bottom-up*. In großen Projekten wird man top-down vorgehen, das heißt, man wird sich zunächst einen groben Überblick über die Problemumgebung verschaffen und sie dann in diverse Module einteilen. Diese werden dann an verschiedene Teams vergeben, die im Laufe der weiteren Analyse ggf. Teilmodule definieren. Im Gegensatz dazu steht die Bottom-Up-Methode: Sie sieht vor, zunächst einzelne Teilbereiche zu modellieren und diese dann zu größeren Modulen zusammenzufassen. Dabei wird gezielt nach semantisch zusammenhängenden Einheiten wie Generalisierungshierarchien oder Aggregationen gesucht. Je nach Bedarf ist es möglich, die beiden Vorgehensweisen zu kombinieren.[5]

Definition von Attributen

Ein *Attribut* ist eine Eigenschaft, die jedes Objekt einer Klasse besitzt. Die von Coad/Yourdon vorgeschlagene OOA sieht für Klassen keine Attribute vor. Klassen besitzen keine Eigenschaften, sondern reichen nur dort definierte Attribute an Subklassen weiter. Bei der Definition der Attribute muß daran gedacht werden, daß ein Objekt mehrere Stadien durchlaufen kann. Solche Statusinformationen über den Lebenszyklus (*life cycle*) [CC90] müssen in einem Attribut gespeichert werden.

Die Auswahl der Attribute richtet sich nach deren Relevanz für die Problemumgebung. Prinzipiell sollte jedes Attribut so weit oben in der Generalisierungshierarchie stehen wie möglich, um die Vorteile der Vererbung weitestgehend ausnutzen zu können. Bei der Verwendung von Hierarchien in ihrer allgemeinen Form kann es durch Mehrfachvererbung zu Namenskonflikten kommen. Besteht der Wert eines Attributs aus Referenzen zu anderen Objekten der Modellwelt, so wird diese Beziehung als *Instance-Connection* bezeichnet und ebenfalls dargestellt. Dadurch lassen sich Abhängigkeiten der Objekte untereinander besser darstellen.

Definition der Methoden

Die Dynamik des Systems wird durch Methoden ausgedrückt. Mit Hilfe von Statusdiagrammen wird der Lebenszyklus jedes Objekts einer Klasse definiert. Die Detailliertheit des Statusdiagramms leitet sich aus Problemumgebung und dem zu modellierenden Zielsystem ab. Alle Attribute, die das Verhalten des Objektes beeinflussen, müssen darin aufgenommen werden. Im Anschluß daran werden die benötigten Methoden festgelegt. Danach wird sukzessive für jede Klasse, für die Methoden vorgesehen sind, geprüft, mit welchen anderen Objekten deren Objekte kommunizieren, d.h. an welche sie Nachrichten versenden. Diese Kommunikationswege werden als *Message-Connections* bezeichnet. Im letzten Schritt werden die Methoden nun genauer spezifiziert mit Ein- und Ausgaben, sowie einem groben Flußdiagramm.

3.2 Objektorientiertes Design

Mit der Durchführung der OOA wurden alle relevanten Aspekte des zu modellierenden Ausschnitts der realen Welt in eine objektorientierte Darstellung übertragen. Das objektorien-

5. Wie wir später schildern werden, haben wir diesen Schritt an den Anfang der Analyse des Teilmodells Produkt gestellt, d.h. wir haben zunächst einen top-down-Entwurf verfolgt, sind aber im Verlauf des weiteren Entwurfs bottom-up vorgegangen.

tierte Design [CY91b] füllt die Lücke zwischen OOA und einer konkreten Implementierung. Terminologie und Notation des OOD sind identisch mit der der OOA, wodurch der von klassischen Entwurfsmethoden bekannte Bruch zwischen Analyse und Design [FK92] vermieden wird.

Coad/Yourdon unterteilen das objektorientierte Design in vier Komponenten:

- Problemumgebungskomponente (Problem-Domain-Component):
  In dieser Komponente wird das Ergebnis der OOA nach Gesichtspunkten der Implementierbarkeit (im gewählten Zielsystem) untersucht und nach Kriterien wie Speicherplatzbedarf optimiert.
- Mensch-Maschine-Schnittstelle (Human-Interaction-Component):
  Hier werden Fenster, Eingabemasken und Ausgaben für die verschiedenen Benutzer des Systems spezifiziert.
- Datenverwaltungskomponente (Data-Management-Component):
  Diese Komponente ist für das Speichern/Laden von Objekten zuständig. Wie diese Komponente im einzelnen aussieht, hängt entscheidend von der Implementierungsumgebung ab (z.B. objektorientierte Programmiersprache oder objektorientiertes DBMS).
- Task-Manager (Task-Management-Component):
  Tasks sind Prozesse, auf die Methoden abgebildet werden. In einem großen System laufen in der Regel Tasks parallel ab (Multitasking), was durch Task-Manager gesteuert wird.

Diese Komponenten werden als zusätzliche Problemfelder betrachtet, die, genau wie die Problemumgebung selbst, analysiert und modelliert werden müssen. Wie detailliert dies geschehen kann, ist allerdings sehr stark von der gewählten Implementierungsumgebung abhängig. Für das in diesem Aufsatz beschriebene Projekt handelt es sich dabei um KRISYS.

4. Überblick über das Wissensbankverwaltungssystem KRISYS

Es gibt drei Klassen von Anforderungen, die an WBVS gestellt werden [Ma91]. Die *effektive Manipulation von Wissen* ist Voraussetzung für eine angepaßte Unterstützung der Anwendungen, die Konstruktion von Wissensbasen erfordert die *flexible Modellierung von Wissen*, und schließlich ist die *effiziente und zuverlässige Verwaltung von Wissen* unabdingbar, um leistungsfähige und sichere Anwendungssysteme erstellen zu können. Aufgrund dieses Aufgabenspektrums läßt sich die Architektur von WBVS konzeptionell in drei Ebenen einteilen, die Anwendungs-, die Entwurfs- und die Implementierungsebene (Bild 2 (a)).

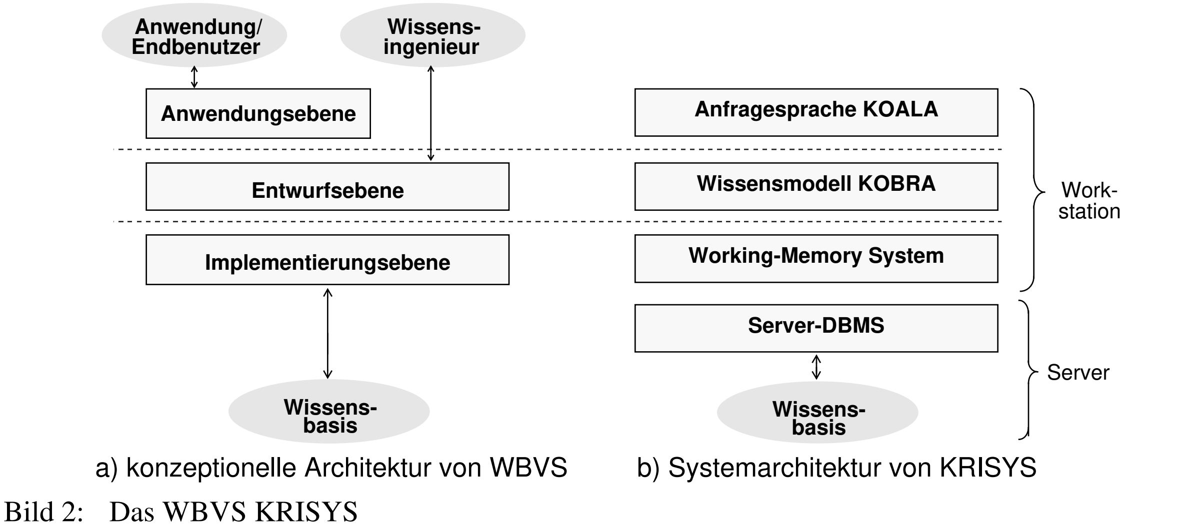


Bild 2: Das WBVS KRISYS

Diese Dreiteilung wurde auch für KRISYS übernommen und in eine Workstation/Server-Architektur übertragen, die eine geeignete Verarbeitungsumgebung für Non-Standard-Anwendungen darstellt [HR85]. Bild 2 (b) gibt einen Überblick über die Systemarchitektur von KRISYS. Innerhalb der Implementierungsebene verläuft die Grenze zwischen Workstation und Server. Auf Serverseite befindet sich ein DBMS, das für einfache Datenverwaltungsaufgaben eingesetzt wird, auf der Workstation ist das Working-Memory-System angesiedelt, das einen Anwendungspuffer bereitstellt, mit dessen Hilfe Workstation und Server lose gekoppelt sind. Die beiden darüberliegenden Ebenen realisieren das Wissensmodell KOBRA und die Anfragesprache KOALA. Da beide im Rahmen dieses Aufsatzes von zentraler Bedeutung sind, werden sie nachfolgend genauer beschrieben.

4.1 KOBRA

Das Wissensmodell KOBRA [Ma91], das auf der Modellierungsebene realisiert ist, unterstützt eine objektzentrierte Darstellung der Anwendungswelt durch den Wissensingenieur. Es erlaubt die Spezifikation deskriptiven, organisationalen und operationalen Wissens in einem einzigen Konzept, dem Objekt. Ein Objekt besitzt einen eindeutigen Namen und eine Menge von Attributen, die seine Eigenschaften beschreiben. Attribute lassen sich einteilen in *Slots*, die zur Darstellung deskriptiven Wissens dienen, und in *Methoden*, mit deren Hilfe operationale Eigenschaften spezifiziert werden können. Um ein Objekt genauer beschreiben zu können, lassen sich für Attribute Integritätsbedingungen, sogenannte Aspekte (z.B. ‘cardinality’, ‘possible-values’, etc.), definieren.

Zur Repräsentation organisationalen Wissens unterstützt KOBRA die Abstraktionskonzepte Generalisierung, Klassifikation, Assoziation und Aggregation [Ma91]. Sie sind in Form spezieller, vordefinierter Slots (z.B. Slot *instance-of* in Bild 3) in das Wissensmodell integriert. Die Semantik der Abstraktionskonzepte, z.B. Vererbung, wird vom System automatisch überwacht und garantiert.

Mercedes-C-180-von-Hr-Maier	
INSTANCE-OF	(PKWs ..)
ELEMENT-OF	(Serienwagen ..)
Km-Stand	(25.000)
possible-values	(interval < 0 2.000.000 >)
cardinality	[1 1]
unit	(km)
demon	(Inspektion_alle_10.000_km)
Anzahl-der-Türen	(4)
default	(4)
hat-Motor	(1.8_l_Motor)
diametric-reference	(Motor-von)
...	

Bild 3: Beispiel für ein Objekt in KRISYS

Neben Methoden unterstützt KOBRA zur Modellierung operationalen Wissens Dämonen und Regeln. Auf diese Weise kann der Wissensingenieur beim Entwurf von Anwendungen auf unterschiedliche Programmierparadigmen zurückgreifen - objektorientierte, datenorientierte und regelbasierte Programmierung. Über Dämonen lassen sich Prozeduren an Attribute anhängen (Aspekt ‘demon’, siehe Bild 3), die, ähnlich Triggern in DBMS, automatisch aktiviert werden, wenn auf die Attribute zugegriffen wird. Regeln werden in Form von Bedingungen (IF-Teil) und Aktionen (THEN-Teil) spezifiziert und mittels KOALA verarbeitet (s. Abschnitt 4.2). Regeln lassen sich in flexibler Weise zu Regelmengen gruppieren, bezüglich derer Inferenzprozesse (Vorwärtsverkettung, Rückwärtsverkettung) durchgeführt werden können. Für beide Ableitungsmechanismen stellt KOBRA vordefinierte Methoden zur Verfügung. Desweiteren kann der Benutzer die Inferenzprozesse über verschiedene Parameter, wie zum Beispiel Konfliktlösungsstrategien, Suchstrategien oder Abbruchbedingungen, steuern. Sowohl

Dämonen als auch Regeln und Regelmengen sind als Objekte in der Wissensbasis (WB) repräsentiert und mit Hilfe der Abstraktionskonzepte organisiert.
Um die zuvor genannten Konzepte bei der Entwicklung von Anwendungen flexibel einsetzen zu können, unterstützt KRISYS die interaktive und schrittweise Konstruktion von Wissensbasen. Jede Entwurfsoperation (z.B. die Definition einer Klasse oder die Reorganisation einer Abstraktionshierarchie) spiegelt sich unmittelbar in der WB wider, so daß der Wissensingenieur seinen Anwendungsentwurf unmittelbar validieren kann, indem er beispielsweise geeignete Methoden oder Inferenzprozesse initiiert.

4.2 KOALA

Als Schnittstelle zu Endbenutzern und Anwendungen besitzt KRISYS die deskriptive, mengenorientierte Anfragesprache KOALA, die zwei Befehle, ASK und TELL, anbietet. Der ASK-Befehl dient zur Abfrage von Wissen und hat folgendes Format:

(ASK <Projektionsteil> <Selektionsteil>): WB $\times$ information $\rightarrow$ information

Durch den Selektionsteil wird eine Qualifikationsbedingung angegeben, die alle Elemente der Ergebnismenge erfüllen müssen. Weiterhin werden im Selektionsteil Anfragevariablen an Werte gebunden. Im Projektionsteil gibt der Anwender an, welche Informationen aus der Ergebnismenge der Selektion herausprojiziert werden sollen.
Die allgemeine Form des TELL-Befehls lautet:

(TELL <Zuweisungsteil> [WHERE <Selektionsteil>]): WB $\times$ information $\rightarrow$ WB

Der Selektionsteil hat den gleichen Aufbau wie bei ASK. Er ist nur vorhanden, wenn im Zuweisungsteil Variablen verwendet werden. Durch den Zuweisungsteil bestimmt der Benutzer oder Wissensingenieur den gewünschten Zielzustand der WB nach Ausführung des Befehls. Diese Zustandsorientierung ist eine wesentliche Eigenschaft des TELL-Befehls. Durchzuführende Manipulationen werden nicht in Bezug auf den aktuellen Zustand der Wissensbasis formuliert (z.B. in Form von *insert*-, *update*- oder *delete*-Operationen), sondern indem der gewünschte Zielzustand spezifiziert wird. Bei der Formulierung von TELL-Ausdrücken muß der Benutzer also nicht den aktuellen Zustand der WB kennen, sondern es bleibt KOALA überlassen, geeignete Maßnahmen zu treffen, um die im TELL genannten Ziele zu erreichen. Mengenorientierte Zusicherungen sind durch Verwendung von Anfragevariablen möglich.
Darüberhinaus erlauben sowohl ASK als auch TELL den Zugriff auf Metainformation, wozu spezielle Prädikate und Funktionen zur Verfügung stehen [DLM90].
Die weiter oben erwähnten Regeln werden intern in TELL-Befehle umgewandelt, wobei der Bedingungsteil einer Regel dem Selektionsteil des TELL entspricht und der Aktionsteil der Regel dem Zusicherungsteil des TELL.

5. Objektorientierte Repräsentation des Teilmodells Produkt

In diesem Kapitel stellen wir die Ergebnisse der objektorientierten Analyse des Teilmodells *Produkt* vor und skizzieren - aus Platzgründen gleichzeitig - deren Umsetzung mit Hilfe von KRISYS. Daher werden Objekte in einer Notation präsentiert, die inhaltlich der von Coad/Yourdon vorgeschlagenen Darstellung entspricht, allerdings bereits die spätere Umsetzung in KRISYS andeutet.
Wie bereits in Abschnitt 3 bemerkt, haben wir, anders als im Ansatz von Coad/Yourdon, die Modularisierung an den Anfang der Analyse gestellt. Diese Vorgehensweise wird ebenfalls bei den STEP-Normierungsbemühungen verfolgt [ISO94]. Auch dort unterteilt man die reale Welt in einzelne Problemfelder, modelliert darin zunächst grundlegende Konzepte (Ressourcen), bevor man eine detailliertere Aufschlüsselung der Problemfelder angeht (Application Protocols). Nach unseren Erfahrungen hat sich gezeigt, daß die nach der top-down-Untergliederung entstandenen Problemfelder nicht unabhängig voneinander modelliert werden können,

sondern daß die bereits abgebildeten Teilausschnitte bei der weiteren Modellierung zu berücksichtigen sind (siehe z.B. Abschnitt 5.1.2). Dabei ist es sinnvoll, zunächst die Abhängigkeiten der Problemfelder untereinander festzustellen. Anschließend wird man zuerst das Problemfeld modellieren, von dem die meisten anderen Problemfelder abhängig sind.

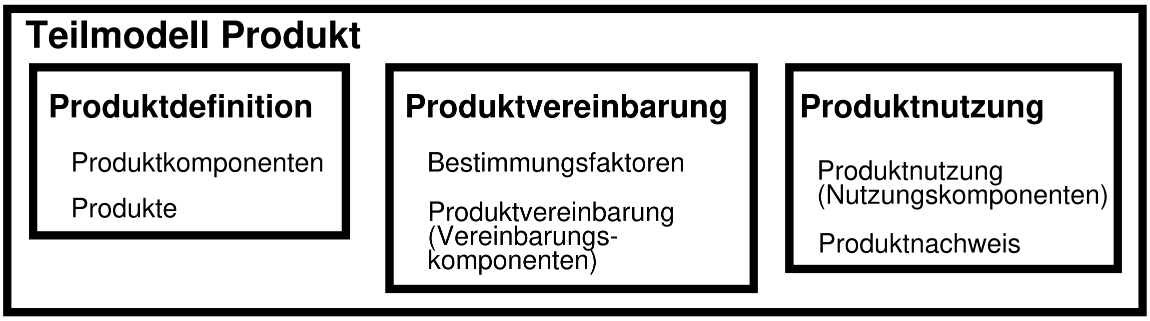


Bild 4: Untergliederung in Problemfelder

Übertragen auf das Unternehmensmodell der R+V-Versicherung ergibt sich dadurch folgendes Vorgehen: das Teilmodell *Produkt* stellt das oberste Problemfeld dar. Dieses kann in die Problemfelder Produktdefinition (wiederum aufgeteilt in Produktkomponenten und Produkte), Produktvereinbarung (zusammengesetzt aus Bestimmungsfaktoren und Vereinbarungskomponenten) und Produktnutzung (bestehend aus Nutzungskomponenten und Produktnachweis) aufgeteilt werden. Die Produktnutzung setzt die Vereinbarung eines Produkts voraus. Diese ist wiederum nur möglich, wenn zuvor ein entsprechendes Produkt definiert wurde. Diese Zusammenhänge bestimmen die Abhängigkeiten zwischen den Problemfeldern. In Bild 4 sind die Problemfelder in der von Coad/Yourdon vorgeschlagenen Notation gezeigt.

5.1 Modellierung der statischen Aspekte des Teilmodells Produkt

Für die gefundenen Problemfelder werden gemäß Coad/Yourdon zunächst Klassen und Objekte definiert, diese strukturiert und durch Attribute genauer spezifiziert. Es wird im folgenden deutlich, daß bereits definierte Objekte in diesen Prozeß mit einzubeziehen sind.

5.1.1 Produktkomponenten

Ein Produkt wird über eine baumartige Komponentenstruktur definiert. Produktkomponenten stellen dabei die einzelnen Bausteine dieser Struktur dar. Produktkomponenten erben ihre Attribute von der Klasse ‘Prodkomp-Klasse’, in der die Struktur aller Produktkomponenten zentral verwaltet wird und folglich auf einfache Weise für alle Objekte geändert werden kann. Das Objekt Prodkomp-Klasse ist in Bild 5 dargestellt.[6]

Bei der Vollkaskoversicherung gibt es beispielsweise die beiden Produktkomponenten ‘Vollkasko’ und ‘Teilkasko’, die beide Instanzen von ‘Produktkomp-Klasse’ sind und über die Aggregationsbeziehungen ‘hat-Prod-Komponente’ bzw. ‘Prod-Komponente-von’ miteinander verbunden sind.

Prodkomp-Klasse	
HAS-INSTANCES	**(Teilkasko Vollkasko Unfallvers ..)**
hat-Prod-Komponente	( )
Prod-Komponente-von	( )

Bild 5: Objekt ‘Produktkomp-Klasse’

Die Mehrfachverwendung von Produktkomponenten kann durch diese Art der Modellierung unterstützt werden. Soll zum Beispiel die Produktkomponente ‘Teilkasko’ ebenfalls zum

6. Aus Platzgründen wird in den folgenden Darstellungen von Objekten immer nur der relevante Ausschnitt des betreffenden Objektes dargestellt.

Umfang eines anderen Produkts gehören, so wird dies durch eine weitere Aggregationsbeziehung zwischen ‘Teilkasko’ und dementsprechenden Produkt erreicht.

5.1.2 Produkte

In der Aufgabenstellung wird gefordert, daß Produkte eine Spezialisierung von Produktkomponenten sein sollen (vgl. Bild 1). Allerdings ist eine Trennung von produkt- und produktkomponentenspezifischer Information wünschenswert, da es Produktkomponenten geben kann, die gleichzeitig als eigenständige Produkte vorgesehen sind. So umfaßt die Vollkaskoversicherung zwar die Leistungen der Teilkaskoversicherung, jedoch soll bei der Bearbeitung des Vollkasko-Produkts nur auf Informationen der Produktkomponente ‘Teilkasko’ zugegriffen werden, nicht aber auf die des Teilkasko-Produkts. Diese dürfen nur dann dem Anwender zugänglich sein, wenn das Teilkasko-Produkt betrachtet wird. Deshalb gibt es für alle Produktkomponenten, die gleichzeitig Produkte sein können, eigene Objekte, die Instanzen von ‘Produkt-Klasse’ sind (s. Bild 6). Beispielsweise existiert für die Produktkomponente ‘Teilkasko’ das Objekt ‘Teilkasko-Produkt’. Das jeweilige “Produkt-Objekt” ist über eine Aggregationsbeziehung mit den entsprechenden Produktkomponenten zu verbinden, d.h., zur Modellierung von Produkten sind immer noch, wie in der Aufgabenstellung verlangt, Produktkomponenten heranzuziehen.

Produkt-Klasse	
HAS-INSTANCES	**(Teilkasko-Produkt Vollkasko-Produkt ..)**
Produkt-von	( )
Kündigung-Frist	( )
Verkauf-Beginn-Datum	( )
Verkauf-Ende-Datum	( )

Bild 6: Objekt ‘Produkt-Klasse’

5.1.3 Produkvereinbarung und Bestimmungsfaktoren

Die Produktvereinbarung dient zur Erfassung aller Daten, die bei einem Vertragsabschluß anfallen.Da ein Vertragsabschluß sich immer auf ein bestimmtes existierendes Produkt bezieht, ist die baumartige Produktomponentenstruktur zusammen mit dem Produkt-Objekt, im folgenden als Produktdefinitionsstruktur bezeichnet, Grundlage für die Produktvereinbarung. Die meisten Informationen, die bei einem Vertragsabschluß zu erfassen sind, können sogenannten Bestimmungsfaktoren zugeordnet werden. Andere Angaben sollen im folgenden außer Betracht gelassen werden.

Bestimmungsfaktoren

Die Komponenten (sowohl Produkt-, wie auch Vereinbarungs- oder Nutzungskomponenten, vgl. nachfolgende Abschnitte) unterscheiden sich jeweils voneinander durch den Versicherungsumfang, für den sie bestimmt sind. So deckt die Vollkaskoversicherung auch den selbstverschuldeten Schaden ab, der bei der Teilkaskoversicherung nicht inbegriffen ist. Demnach muß es möglich sein, für jede Komponente deren entsprechenden Versicherungsumfang zu repräsentieren. Dieser Umfang wird durch die drei sogenannten *Bestimmungsfaktoren* ‘Ereignis’, ‘Schaden’ und ‘Leistung’ festgelegt. Neben diesen Bestimmungsfaktoren, die gruppiert den Leistungsumfang definieren, gibt es auch solche, die jeweils einzeln Vereinbarungskomponenten zugeordnet werden können. So ist bei einem Vertragsabschluß beispielsweise ein ‘Sachbearbeiter’, der ‘Versicherungsnehmer’, etc. zu nennen. Um die Bestimmungsfaktoren voneinander unterscheiden zu können, werden sie durch eigene Klassen repräsentiert. Konkrete Bestimmungsfaktoren, die sogenannten *Bestimmungsfaktor-Objekte*, sind Instanzen dieser Klassen.

Bestimmungsfaktoren haben also zwei Aufgaben zu erfüllen, zum einen die detaillierte Beschreibung einer Produkt- und Vereinbarungskomponente, zum anderen die Festlegung des Leistungsumfangs für eine Produktkomponente. Nachfolgend wird die Modellierung von Bestimmungsfaktoren anhand dieser beiden Aufgaben besprochen.

Zur detaillierten Beschreibung einer Vereinbarungskomponente gehören u.a. der Name des Sachbearbeiters oder der Geltungsbereich der Vereinbarung. Entsprechend existieren die Bestimmungsfaktor-Klassen *Sachbearbeiter* und *Geltungsbereich*, sowie zum Beispiel das Bestimmungsfaktor-Objekt *Sachbearbeiter_Hr_Maier*.

Die Bestimmungsfaktor-Objekte, die bei einem Vertragsabschluß für ein bestimmtes Produkt zu verwalten sind, werden durch Aggregationsbeziehungen mit den entsprechenden Vereinbarungskomponenten verbunden. Dabei gibt es für verschiedene Bestimmungsfaktoren-Arten auch unterschiedliche Aggregationsattribute.

Beispiel: Bei einer Teilkaskoversicherung sollen u.a. der Sachbearbeiter und der Geltungsbereich aufgeführt werden. Folglich bestehen jeweils zwischen Bestimmungsfaktor-Objekten der Klassen Sachbearbeiter und Geltungsbereich sowie der Vereinbarungskomponente Teilkasko-Vereinbarung Aggregationsbeziehungen (dargestellt durch die Attribute 'Sachbearbeiter' bzw. 'Geltungsbereich').

Die Aggregationsattribute werden in sogenannten *Beziehungsobjekten* definiert und durch Instanziierung an die entsprechenden Vereinbarungs-Komponenten vererbt. Die Beziehungsobjekte haben den ausschließlichen Nutzen, die Aggregationsattribute, welche Bestimmungsfaktor-Objekte mit Komponenten verbinden, zu definieren.

Beispiel: Bei einer Teilkasko-Vereinbarung ist als Bestimmungsfaktor das versicherte Objekt anzugeben. Also muß die Teilkasko-Vereinbarung Instanz des Beziehungsobjektes sein, in dem das Aggregationsattribut 'versichertes-Objekt' definiert ist. Dieses Attribut wird durch die Instanziierung an 'Teilkasko-Vereinb' vererbt, so daß eine Beziehung zwischen 'Teilkasko-Vereinb' und dem entsprechenden Bestimmungsfaktor hergestellt werden kann (vgl. Bild).[7]

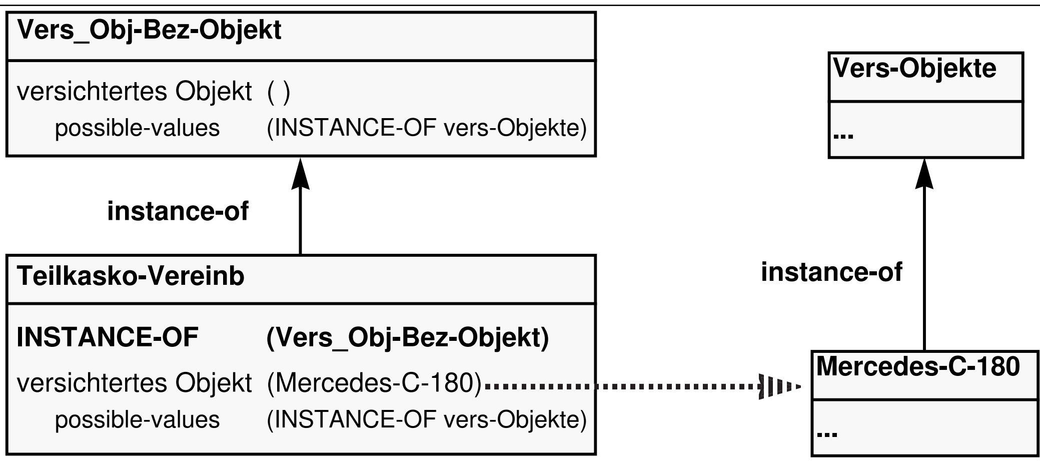


Bild 7: Auswirkung der Instanziierung eines Bezieungsobjektes

Durch die Beziehungsobjekte wird eine zentrale Verwaltung der Aggregationsattribute und der für sie geltenden Integritätsbedingungen möglich. Zum besseren Verständnis wird in Bild 8 ein solches Beziehungsobjekt dargestellt.

Der Leistungsumfang für ein Produkt muß bei der Versicherungsvereinbarung vorgenommen werden. Er legt die Versicherungsleistungen fest, die für einen bestimmten Schaden, der wiederum durch ein gewisses Ereignis eingetreten ist, erbracht werden. Die Bestimmungsfaktoren 'Schaden', 'Ereignis' und 'Leistung' sind also, wie zu Beginn des Abschnitts erwähnt, zu

7. Dieses Bild zeigt schematisch die Instanziierung eines Beziehungsobjektes. In Bild 10 ist die korrekte Beziehung zwischen Vereinbarungskomponenten und Beziehungsobjekten dargestellt.

einer Gruppe zusammenzufassen. Folglich werden die entsprechenden Bestimmungsfaktoren jeweils in einem *Paket* gruppiert, das als eigenständiges Objekt repräsentiert ist. Ein Paket und dessen zugehörige Bestimmungsfaktoren werden über Aggregationsbeziehungen verbunden. Ein Paket muß folglich Instanz der entsprechenden Beziehungsobjekte sein, um die jeweiligen Aggregationsattribute zu den entsprechenden Bestimmungsfaktoren zu erben.

```
Ereignis-Bez-Objekt
HAS-INSTANCES         (Verlust-durch-Diebstahl-Paket ..)
HAS-SUBCLASSES        ( .. )
ELEMENT-OF            (Beziehungsobjekte)
Ereignis              ( )
   diametric-reference   (Ereignis-von)
   possible-values       (INSTANCE-OF Ereignis-Klasse)
   cardinality           [1 1]
```

Bild 8: Beispiel für ein Beziehungsobjekt

Beispiel: *Für eine Teilkaskoversicherung soll der Schaden “Verlust” durch das Ereignis “Diebstahl” versichert werden können, wobei als Leistungen “Anwalt-”, “Mietwagen-” und “Wiederbeschaffungskosten” abgedeckt sind. Das Paket zu diesem Beispiel ist in Bild 9 gezeigt (‘Auto-Diebstahl’, ‘Auto-Verlust’, ‘Anwalt-Leistung’, ‘Mietwagen-Leistung’ und ‘Wiederbeschaffungs-Leistung’ sind die entsprechenden Bestimmungsfaktor-Objekte):*

```
Verlust-durch-Diebstahl-Paket
INSTANCE-OF        (Ereignis-Bez-Objekt Schaden-Bez-Objekt Leistung-Bez-Objekt)
Ereignis           (Auto-Diebstahl)
   possible-values    (AND  (INSTANCE-OF Ereignis-Klasse)
                            (INSTANCE-OF Diebstähle))
   cardinality        [1 1]
Schaden            (Auto-Verlust)
   possible-values    (AND  (INSTANCE-OF Schaden-Klasse)
                            (INSTANCE-OF Verlust-Schäden))
   cardinality        [1 1]
Leistung           (Anwalt-Leistung Mietwagen-Leistung Wiederbeschaffungs-Leistung)
   possible-values    (AND  (INSTANCE-OF Leistung-Klasse)
                            (OR  (INSTANCE-OF Anwalt-Leistungen)
                                 (INSTANCE-OF Mietwagen-Leistungen)
                                 (INSTANCE-OF Wiederbeschaffungs-Leistungen)))
   cardinality        [1 3]
```

Bild 9: Objektstruktur eines Paketes

Produktvereinbarung

Eine Produktvereinbarung entsteht durch Instanziierung einer Produktdefinitionsstruktur. Dadurch werden alle notwendigen Attribute automatisch in die Produktvereinbarung übernommen. Wie bereits erwähnt, werden Bestimmungsfaktoren durch Aggregationsattribute, die in Beziehungsobjekten definiert sind, mit Vereinbarungskomponenten verbunden. Da bereits bei der Produktdefinition festgelegt werden soll, welche Bestimmungsfaktoren von einer Vereinbarungskomponente aus referenziert werden dürfen, ist die entsprechende Produktkomponente Subklasse des Beziehungsobjektes (vgl. Bild 10).

Beispiel: *Bei der Definition des Produkts ‘Teilkasko’ wird festgelegt, daß bei einem später abzuschließenden Vertrag ein Sachbearbeiter, ein Geltungsbereich, etc. aufzunehmen ist. Dadurch, daß die Teilkasko-Produktkomponente (in Bild 10 ‘Teilkasko’) Subklasse des Beziehungsobjektes ist (‘Geltungsbereich-Bez-Objekt’) erbt sie das Aggregationsattribut ‘Geltungsbereich’. Dieses wird bei der Instanziierung der Produktkomponenten (während eines Vertragsabschlusses) an die entsprechende Vereinbarungskomponente weitervererbt. Somit kann bei einem Vertrag*

(‘Teilkasko-Vereinb’) der Geltungsbereich über das geerbte Aggregationsattribut ‘Geltungsbereich’ aufgeführt werden.

Eine Produktaktion ist dafür verantwortlich, daß die Produktstruktur korrekt auf die Vereinbarungsstruktur übertragen wird, was allein durch die Instanziierung von Produktkomponenten nicht möglich ist. Auf Produktaktionen wird in Abschnitt 5.2 eingegangen.

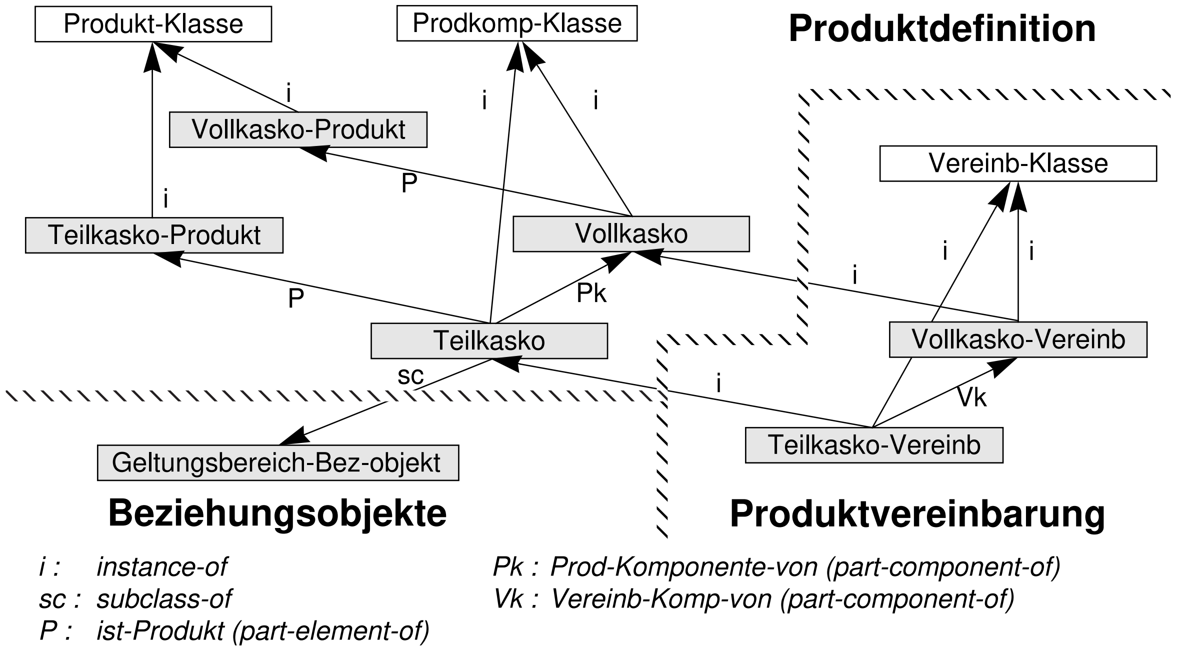


Bild 10: Zusammenhang zwischen Produktdefinition und -Vereinbarung

5.1.4 Produktnutzung

Im Rahmen der Produktnutzung werden Daten erfaßt, die bei einem bestimmten Vorfall aufgetreten sind. Sie betreffen die Bestimmungsfaktor-Pakete: welches Ereignis ist eingetreten (z.B. ein Diebstahl), zu welchem Schaden (z.B. dem Verlust) hat es geführt usw. Anhand der Pakete, die in dem Vereinbarungsbaum aufgeführt sind, ist das Paket auszuwählen, das zu dem Vorfall “paßt”, d.h. zur Aufnahme der Daten verwendet werden kann.

Im Gegensatz zur Vereinbarung muß bei der Nutzung nicht mehr der vollständige Produktbaum übertragen werden, denn in den meisten Fällen wird, wie beispielsweise bei einer Teilkaskoversicherung, nicht der volle Umfang des Produkts in Anspruch genommen. Daher wäre es ungünstig, bei der Produktnutzung den vollständigen Produktbaum verwalten zu müssen. Ebenso wie bei Produkt- und Vereinbarungskomponenten wird auch die Struktur von Nutzungskomponenten in einem Objekt (‘Nutzung-Klasse’) definiert. Als Modellierungskonstrukt für den Nutzungsbaukasten wird, wie bisher auch, die Aggregation verwendet (vgl. Bild 11).

Beispiel: Im Rahmen einer Teilkaskoversicherung sei ein Verlust durch Diebstahl abzuwickeln. Hierzu wird das ‘Verlust-Diebstahl-Vereinb-Paket’ als Vorlage ausgewählt, da es sich zur Aufnahme der Daten am besten eignet. Indem eine Instanz des relevanten Bestimmungsfaktor-Pakets der Vereinbarung (also von ‘Verlust-Diebstahl-Vereinb-Paket’) erzeugt wird, kann zwischen dem generierten Nutzungspaket (‘Teilkasko-Nutzung-01-01-94’) und den Bestimmungsfaktoren, die den Vorfall betreffen (‘Auto-Diebstahl’, ‘Auto-Verlust’ und ‘Anwalt-Leistung’), über das Nutzungspaket (‘Verlust-durch-Diebstahl-Paket’) eine Beziehung erzeugt werden.

Bisher kann auf alle Nutzungskomponenten über das Objekt ‘Nutzung-Klasse’ zugegriffen werden. Es ist jedoch erwünscht, alle Nutzungen mit der Vereinbarung und dem Produkt zusammenzufassen. Dies wird durch den Produktnachweis ermöglicht.

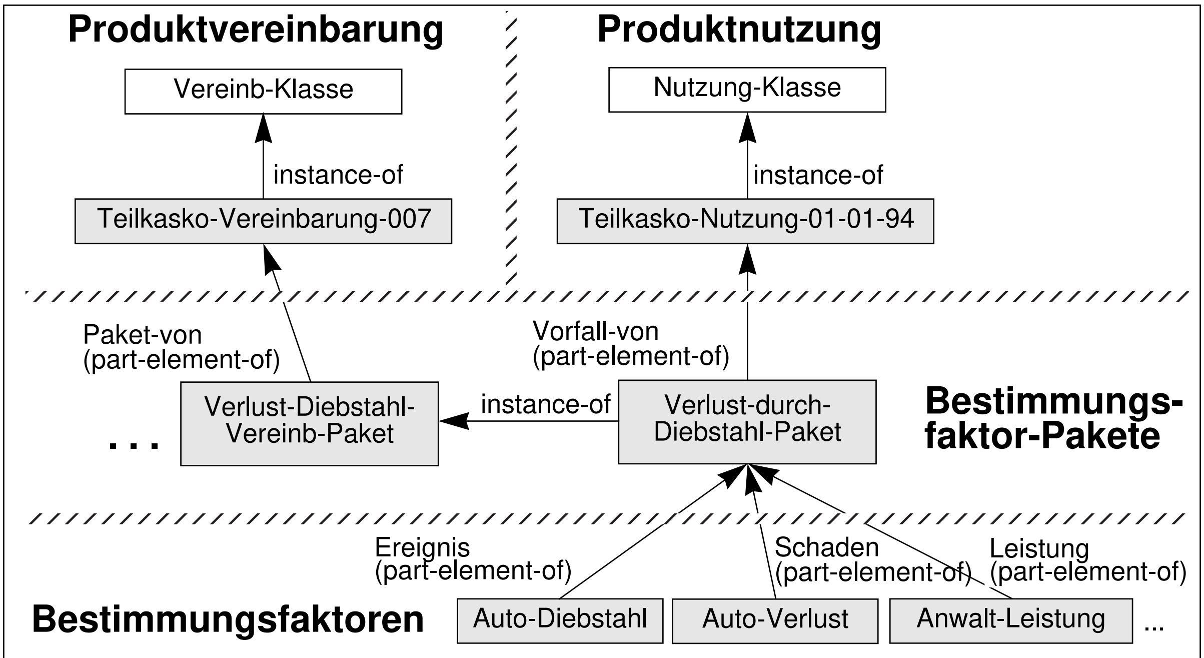


Bild 11: Verwaltung der Daten zu einem Vorfall

5.1.5 Produktnachweis

Es gibt zwei Alternativen, um die Zusammenfassung aller Nutzungen, der Vereinbarung und des Produkts zu modellieren: die Assoziation und die Aggregation. Werden alle Komponenten durch eine Menge miteinander verbunden, so entsteht der Nachteil, daß man von der Beziehung her keine Unterscheidung zwischen dem Produkt, der Vereinbarung und den Nutzungen hat. In KRISYS werden alle Elemente einer Menge über das gleiche Beziehungsattribut mit dem Mengenobjekt verbunden. Eine semantische Unterscheidung der Elemente von dem Mengenobjekt aus ist dadurch nicht möglich. Deshalb wurde die Aggregation zur Modellierung des Nachweises verwendet. Dadurch können das Produkt zu dem Nachweis, die Vereinbarung und die Nutzungen durch jeweils eigene Aggregationsattribute mit dem Nachweisobjekt verbunden werden. Dies trägt zur besseren Unterscheidung bei. Wie bei Produkten, Produkt-, Vereinbarungs- und Nutzungskomponenten wird die Struktur für alle Nachweise in einem Objekt (‘Nachweis-Klasse’) definiert.

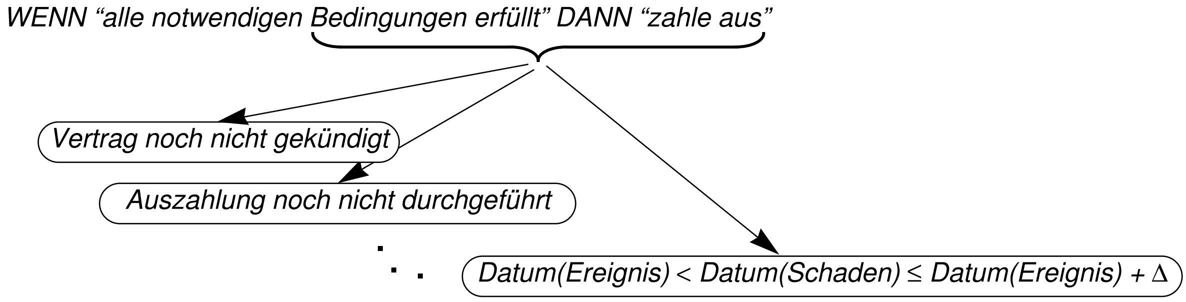
5.2 Modellierung der Dynamik

Dadurch, daß wir die Modularisierung an den zu realisierenden Methoden orientiert haben, konnten alle von einer Methode benötigten Informationen zusammenhängend dargestellt werden, da sie sich innerhalb eines Problemfelds befinden.

Dynamische Aspekte des Teilmodells Produkt sind der Aufbau der Produkt-, Vereinbarungs- und Nutzungsstrukturen. Da hierbei oft objektübergreifende Zugriffe durchzuführen waren, konnten die Methoden nicht immer einzelnen Objekten zugeordnet werden. Aus diesen Gründen modellierten wir die anwendungsspezifischen Methoden in eigenen Objekten, je eines für Produktgestaltung, Vertragsabschluß und Produktnutzung. Als Konsequenz daraus ergab sich eine zentrale Verwaltung aller Anwendungsprogramme mit einer dadurch verbundenen leichteren Wartbarkeit. Die Methoden wurden so implementiert, daß der Benutzer die Eingabe von Daten unterbrechen (beispielsweise wenn Informationen für die weitere Bearbeitung fehlen) und die Methode zu einem späteren Zeitpunkt neu aufrufen kann.

Wie bereits in der Einleitung von Abschnitt 5 angedeutet wurde, stellt KOALA die eigentliche Schnittstelle zwischen Benutzer und WBVS dar. Da der Aufbau der Produkt-, Vereinbarungs- und Nutzungsstrukturen jedoch sehr komplex ist, sollte der Benutzer durch eine geeignete Dialogführung unterstützt werden.

Die anwendungsspezifischen Methoden durchwandern rekursiv die zu bearbeitenden Baukästen (Produktkomponenten, Vereinbarungskomponenten, Nutzungskomponenten). Durch eine ständig aktualisierte grafische Repräsentation des gerade zu bearbeitenden Ausschnitts der Daten wird der Benutzer bei der Erzeugung komplexer Strukturen unterstützt. Desweiteren bieten die Methoden dem Benutzer nur genau die Attribute zur Wertangabe an, die für diesen Bearbeitungsvorgang relevant sind. Welche Attribute das im Einzelnen sind, ist nicht in den Methoden codiert, sondern in den Objekten spezifiziert, was eine bessere Lesbarkeit und Wartbarkeit ermöglicht.
Im Gegensatz zu allen anderen Produktaktionen ist die (versuchte) Inanspruchnahme durch Regeln realisiert. Der Hauptgrund hierfür ist die allgemeine Form der Inanspruchnahme:



Die Bedingungen werden jeweils durch Regeln spezifiziert. Weil diese zwischen verschiedenen Produkten stark variieren können, werden Regeln mittels produktbezogener Regelmengen zusammengefaßt. Da Regeln in mehreren Regelmengen vorhanden sein können, wird die Mehrfachverwendung der gleichen Bedingung in mehreren Produkten unterstützt. Desweiteren kann die Regelmenge eines bereits existierenden Produkts (z.B. Teilkasko) in die Regelmenge des neuen Produkts (z.B. Vollkakso) aufgenommen werden.

6. Zusammenfassende Bewertung

In den vorangehenden Abschnitten wurde die objektorientierte Analyse der Informationsverarbeitung der R+V-Versicherung am Beispiel des Teilmodells *Produkt* diskutiert und deren beispielhafte Implementierung mit Hilfe des WBVS KRISYS beschrieben. Abschluß und wesentlicher Bestandteil des Projekts war die Validierung der so erzielten Ergebnisse hinsichtlich ihrer Adäquatheit für die Erfordernisse der R+V-Versicherung. Ein weiterer Schwerpunkt war die Realisierbarkeit mittels eines am Markt verfügbaren DBMS, da - wie schon in der Einleitung erwähnt - KRISYS als prototypisches System nicht den Leistungsanforderungen großer kommerzieller Anwendungen gewachsen ist. Im folgenden fassen wir die Ergebnisse dieser Bewertung zusammen.
Die objektorientierte Analyse erlaubte eine exakte und natürliche Beschreibung sowohl der statischen Aspekte als auch der dynamischen Aspekte der zu modellierenden realen Welt. Viele Details des Teilmodells *Produkt*, für die im vorgegebenen ER-Diagramm entweder gar keine oder nur eine sehr umständliche Darstellung möglich ist, konnten auf diese Weise berücksichtigt werden. Dies wird schon auf den ersten Blick deutlich, wenn man das ursprüngliche ER-Diagramm und die daraus entstandenen Fülle an Klassen, Mengen und Aggregaten vergleicht. Desweiteren fehlen dynamische Aspekte des Teilmodells im ER-Diagramm völlig. Das Ergebnis der objektorientierten Analyse spiegelte dementsprechend gut die Eigenschaften des Teilmodells *Produkt* wider und entsprach damit auch voll den zu Anfang des Projekts gestellten Anforderungen der R+V-Versicherung (vgl. Abschnitt 2).
Bei den Untersuchungen zum objektorientierten Design, d.h. der praktischen Umsetzung der Ergebnisse der OOA, offenbarten sich Probleme, die zum einen mit der gewählten Analyse- und

Entwurfsmethodik zusammenhängen, zum anderen durch die Fähigkeiten kommerziell verfügbarer Systeme bedingt sind.

Die aus der OOA resultierende Modellierung enthält eine Vielzahl von Klassen (für das gesamte Teilmodell 257 Klassen) mit einer zum Teil potentiell geringen Anzahl Instanzen. Zwar erfüllt diese Darstellung die an sie gestellten Anforderungen, insbesondere hinsichtlich Flexibilität, es ist aber fraglich, ob eine solch diversifizierte Modellierung sich in ein effizientes Anwendungssystem umsetzen läßt. Dieser Gegensatz zwischen Flexibilität einerseits und Effizienz andererseits ist grundsätzlicher Natur und hängt zunächst einmal nicht mit dem später zur Implementierung verwendeten realen System zusammen. Es erscheint uns daher durchaus sinnvoll, zu Beginn des objektorientierten Designs die Ergebnisse der OOA dahingehend zu untersuchen, wieviel Flexibilität zugunsten einer kompakteren Darstellung aufgegeben werden kann bzw. soll. Dieser Aspekt wird jedoch in vielen Ansätzen zur OOD nur unzureichend berücksichtigt [FK92].

Eine Schwäche des OOD, die auch in [FK92] kritisiert wird, sind die fehlenden Modellierungsmöglichkeiten für längerfristige, mehrere Objekte involvierende Prozesse. Ein Beispiel für einen solchen Prozeß ist die Bearbeitung eines Schadensfalls im Teilmodell *Produkt*. Die Bearbeitung ist eine Tätigkeit, die verschiedene Objekte betrifft und deren einzelne Bestandteile zwar mittels des OOD beschrieben werden können (z.B. das Erzeugen eines Bestimmungsfaktors 'Totalschaden', was der Generierung einer Instanz der Bestimmungsfaktor-Art 'Auto-Schaden' entspricht), die aber in ihrer Gesamtheit nicht darstellbar ist. Weil solche Methoden Instanzen mehrerer Klassen erzeugen, lassen sie sich nicht genau einem Objekt oder einer Klasse zuordnen. Eine entsprechende Erweiterung der Konzepte zum OOD wäre wüschenswert.

Betrachtet man die Ausdrucksmächtigkeit heute verfügbarer, kommerzieller OODBMS [Ca91], so wird deutlich, daß die Vielfalt objektorientierter Konzepte aus OOA und OOD mit keinem System vollständig abbildbar sind. So unterstützt beispielsweise kein System alle Abstraktionskonzepte, und viele Systeme bieten nur sehr eingeschränkte Möglichkeiten, Integritätsbedingungen zu formulieren. Aus diesem Grund läßt sich die für das Teilmodell *Produkt* gefundene Modellierung nicht direkt mit einem heute verfügbaren kommerziellen OODBMS umsetzen. Da dieses Teilmodell repräsentative Eigenschaften aufweist, kann diese Aussage wohl auf die gesamte Informationsverarbeitung der R+V-Versicherung ausgedehnt werden.

Für die Restrukturierung der Informationsverarbeitung bieten sich somit zwei Möglichkeiten. Die Maßnahmen könnten zum einen auf solche Bereiche beschränkt werden, die durch die Funktionalität eines heutigen kommerziellen OODBMS abgedeckt werden. Zum anderen könnte man eine objektorientierte Analyse der gesamten Informationsverarbeitung vornehmen und die praktische Umsetzung der so entstehenden, umfassenden Spezifikation an den Möglichkeiten heute verfügbarer OODBMS orientieren. Letztere Alternative hat den Vorteil, daß eine vollständige, in sich konsistente Strukturierung der Informationsverarbeitung vorliegt, die als mittel- bis langfristige Perspektiven dienen kann. So könnte bei Verfügbarwerden semantisch mächtigerer Systeme sofort durch eine entsprechend weitergehende Umsetzung der Ergebnisse der OOA reagiert werden.

Nach unseren Erfahrungen ist die OOA eine geeignete Methode zur Restrukturierung der betrieblichen Informationsverarbeitung, die es erlaubt, alle relevanten Informationen und Aktionen exakt und einheitlich zu modellieren. Zwar muß eine praktische Umsetzung der Analyseergebnisse darauf abzielen, den Benutzern eine uniforme Sicht auf die zu bearbeitenden Daten zu bieten, unsere Untersuchungen bezüglich einer Implementierung haben jedoch gezeigt, daß eine ausschließlich objektorientierte Realisierung aus Effizienzgründen nicht wünschenswert ist. Stattdessen wäre es im Falle der R+V-Versicherung sinnvoll, die Daten je

nach ihrer Struktur und Verwendung unterschiedlich zu verwalten: einfach strukturierte, große Datenmengen, auf denen nur elementare Operationen (wie Einfügen, Löschen oder Ändern von Einträgen) auszuführen sind, könnten in einem relationalen DBMS abgelegt werden, komplex strukturierte Daten mit operationalen Eigenschaften dagegen in einem OODBMS.
Solche Anforderungen können von einem föderierten Datenbanksystem erfüllt werden, in denen sich heterogene DBMS in einem globalen Schema zusammenfassen lassen. Eine Alternative bieten objektorientiert-relationale Systeme, wie sie von [SQL3] vorgeschlagen werden, die die Vorteile der relationalen und objektorientierten Repräsentationen verbinden.

Literatur

Bo94 Booch, G.: Object-Oriented Analysis And Design, Benjamin Cummings, New York, 1994.

BS93 Brodie, M., Stonebraker, M.: Incremental Migration of Legacy Data Base Applications, GTE Laboratories, Waltham, Mass., Technical Report 93-12, Januar 1993.

Ca91 Cattell, R. (ed.): Next Generation Database Systems, in: Special issue of Communications of the ACM, Vol. 34, No.10, 1991.

CC90 Chikofsky, E., Cross, J.: Reverse Engineering and Design Recovery: A Taxonomy, IEEE Software 7 (1), Januar 1990, 13-17.

CY91a Coad, P., Yourdon, E.: Object-Oriented Analysis, Englewood Cliffs 1991.

CY91b Coad, P., Yourdon, E.: Object-Oriented Design, Englewood Cliffs 1991.

DLM90 Deßloch, S., Leick, F.J., Mattos, N.M.: A State-oriented Approach to the Specification of Rules and Queries in KBMS, ZRI-Bericht 4/90, Universität Kaiserslautern, 1990.

DLMT93 Deßloch, S., Leick, F.J., Mattos, N.M., Thomas, J.: The KRISYS Project - A Summary of What We have Learned so far, in: Stucky, W., Oberweis, A. (eds.): Datenbanksysteme in Büro, Technik und Wissenschaft,, Springer (Informatik Aktuell), 1993, 124-143.

EKPR92 Eicker, S., Kurbel, K., Pietsch, W., Rautenstrauch, C.: Einbindung von Software-Altlasten durch integrationsorientiertes Reengineering, Wirtschaftsinformatik, 34. Jahrgang, Heft 2, April 1994, 137-145.

FK92 Fichman, R., Kemerer C.: Object-Oriented and Conventional Analysis and Design Methodologies, IEEE Computer 1992 Vol. 25, No. 10, 22-39.

HR85 Härder, T., Reuter, A.: Architektur von Datenbanksystemen für Non-Standard-Anwendungen, in: Proc. GI-Proc. GI Conf. on Database Systems for Office, Engineering and Scientific Applications, p.253-286, Karlsruhe, März 85, IFB 94, Springer Verlag, Heidelberg.

ISO94 ISO 10303-1: Product Data Representation and Exchange - Part 1: Overview and Fundamental Principles, ISO TC184 / SC4 N193, 1994.

Ma91 Mattos, N.M.: An Approach to Knowledge Base Management - Requirements, Knowledge Representation, and Design Issues -, Lecture Notes in Artificial Intelligence, Vol. 513 , Springer, 1991.

RS94 Radeke, E., Scholl, M.: Federation and Stepwise Reduction fo Database Systems, Proc. International Conference on Applications of Databases, Schweden, Juni 1994.

R+V92 R+V-Versicherung: Unternehmensmodell, Abschlußbericht, Wiesbaden, 1992.

Sa94 Sauter, G.: Modellierung eines flexiblen Baukastens für Versicherungsprodukte mit dem Wissensbankverwaltungssystem KRISYS, Diplomarbeit, Fachbereich Informatik, Universität Kaiserslautern, 1994.

SL90 Sheth, A.P. , Larson, J.A.: Federated Database Systems for Managing Distributed, Heterogeneous, and Autonomous Databases, ACM Computing Surveys, Vol. 22 , No. 3, September 1990, 183-236.

SQL3 ISO working draft Database Languages - SQL3, Februar 1993.

ST93 Stein, W.: Objektorientierte Analysemethoden - ein Vergleich, Informatik-Spektrum (1993) 16.

TK78 Tichritzis, D., Klug, A.: The ANSI/X3/SPARC DBMS Framework Report of the Study Group on Databasemanagement Systems, Information Systems, Vol. 3, 1978