in: ITG-Fachbericht 137, Softwaretechnik in Automation und Kommunikation (STAK'96), München, März 1996, S. 111-126.

RITA - ein rechnergestütztes Informationssystem für technische Anwendungen

Theo Härder, Joachim Thomas

Universität Kaiserslautern, FB Informatik
67653 Kaiserslautern
e-mail: {haerder | thomas}@informatik.uni-kl.de

Kurzbeschreibung:

In diesem Aufsatz werden die Ziele und Lösungsansätze beschrieben, die in einem Kooperationsprojekt mit der Entwicklungsabteilung eines Herstellers von Fahrzeugsitzen[1] aller Art erarbeitet wurden. Ziel der Kooperation ist die prototypische Erstellung eines rechnerbasierten Informationssystems für technische Anwendungen (RITA) zur Unterstützung der Arbeitsabläufe in von Ingenieurarbeit geprägten Systemumgebungen. Neben der DB-Modellierung der dabei anfallenden Daten und Operationen ist die Steuerung und Überwachung der Arbeitsabläufe im Rahmen der Entwurfsprojekte von besonderer Bedeutung. Wir diskutieren beide Aspekte exemplarisch. Daneben zeigen wir auf, welche Softwaretechnologien geeignet sind, um RITA auf der Basis kommerzieller Softwareprodukte zu realisieren.

1. Einführung

Ausgangspunkt für das Kooperationsprojekt war die Beobachtung, daß das in einem Betrieb vorhandene Wissen (technische Informationen, aus früheren Projekten gewonnene Erfahrungen) nicht zentral verwaltet wird und somit nicht für alle Entscheidungsträger gleichermaßen zugänglich ist. Daten und Erfahrungen früherer Versuche sind oft entweder in Akten abgelegt oder sind an Personen gebunden. Der wesentliche Nachteil dieser Form der Datenhaltung ist, daß tatsächlich vorhandenes Wissen bei der Vorbereitung und Durchführung neuer Projekte entweder gar nicht oder nur lückenhaft Berücksichtigung findet. Dies führt zu längeren Entwicklungszeiten, teuren Planungsfehlern und somit insgesamt zu erhöhten Kosten. Die Personenbindung von Wissen stellt einen weiteren, nicht zu unterschätzenden Kostenfaktor dar. Man braucht nur an das Ausscheiden eines Ingenieurs zu denken, dessen Nachfolger sich das verlorengegangene Know-How selbst erarbeiten muß.

Ziel des in diesem Aufsatz beschriebenen Projekts ist die prototypische Erstellung eines rechnerbasierten Informationssystems für technische Anwendungen zur Unterstützung der Arbeitsabläufe in Planung, Konstruktion und Produktion. Primäre Aufgabe des Informationssystems soll es sein, Daten und Erfahrungen früherer Versuche zu verwalten und in geeigneter Form bereitzustellen, um so die genaue und umfassende Vorbereitung von Projekten optimal zu unterstützen. Außerdem soll das System laufend relevante Informationen neuer Projekte in das bereits vorhandene Wissen integrieren. Durch die Einrichtung eines solchen Informationssystems soll ein umfassender Informationsaustausch zwischen den an den Versuchen beteiligten Fachleuten ermöglicht werden, wodurch Entwicklungszeiten verkürzt, die Zielerreichung sicherer und

1. Bereich Zentralversuch der Keiper Recaro. Die Aktivitäten der Firma umfassen alle Arbeitsbereiche, von der Konstruktion neuer Sitze bis zu deren Produktion.

Entwicklungsaufwand reduziert werden sollen. Eine wichtige Zielsetzung bei der Entwicklung von RITA ist die nahtlose Integration des Systems in bereits bestehende Arbeits- und Entscheidungsabläufe. RITA soll die Experten bei der Entscheidungsfindung unterstützen, es soll jedoch nicht den Experten ersetzen.

Im folgenden werden zunächst die konkrete Problemstellung im Bereich Zentralversuch und die dort relevanten Arbeitsabläufe, Aktivitäten und Informationen skizziert. Anschließend beschreiben wir, wie Daten und Funktionen bei der Erfassung von Kundenanforderungen, dem ersten Schritt bei der Durchführung eines Versuchs, in eine objektorientierte Modellierung überführt werden können. Anhand dieser Beschreibung soll beispielhaft aufgezeigt werden, wie die in einzelnen Teilschritten anfallenden Informationen und Operationen dargestellt werden können. Im Anschluß daran beschäftigen wir uns mit Anforderungen an ein Workflow-Management-System, die sich sich aus den von RITA zu kontrollierenden Aktivitäten ergeben. Im vierten Abschnitt dieses Aufsatzes diskutieren wir Realisierungsmöglichkeiten der Datenhaltung für RITA basierend auf kommerziellen Softwareprodukten. Neben der reinen Datenhaltung liegt eine weitere wichtige Aufgabe von RITA in der Steuerung und Überwachung der Arbeitsabläufe im Rahmen eines Projekts.

Zum Abschluß dieses Aufsatzes zeigen wir, wie einige der zuvor erläuterten Modellierungsansätze mit Hilfe des objekt-relationalen Datenbanksystems UniSQL [6, 9] programmiert wurden.

2. Typische Arbeitsabläufe

Ein Projekt zur Entwicklung eines serienreifen Produktes (z. B. Fahrzeugsitz) zieht sich in der Regel über mehrere Monate bzw. Jahre hin und durchläuft dabei mehrere Stadien. Im folgenden werden die prinzipiellen Phasen eines Projekts im Bereich Versuch beschrieben.

2.1 Anforderungskatalog

Jedes Projekt beginnt zunächst mit der Erstellung eines Anforderungskatalogs, dessen zentralen Bestandteil die spezifischen Wünsche des Kunden ausmachen. Diese werden in der Regel von den Kunden in Form eines gut strukturierten Pflichtenhefts bereitgestellt. Eine solche Spezifikation deckt dabei alle für einen Kunden wichtigen Kriterien ab, d. h., es existiert pro Kunde genau ein Pflichtenheft. Abhängig vom Einsatzbereich des späteren Produkts müssen daraus die für den aktuellen Versuch relevanten Teile extrahiert werden. Neben den Wünschen der Kunden sind aber auch gesetzliche Vorschriften sowie firmeninterne Normen relevant. Da diese ebenfalls vom Einsatzbereich des zu entwickelnden Produktes abhängen, sind auch hier, genau wie bei den Kundenanforderungen, nur bestimmte Vorschriften für den zu planenden Versuch von Bedeutung.

Am Ende dieser ersten Phase liegt ein Lastenheft vor, das die Grundlage für alle weiteren Tätigkeiten bildet.

2.2 Prototypische Lösung

Der nächste Schritt im Rahmen eines Versuchs ist die Erstellung eines prototypischen Produktes. Dieses wird, soweit möglich, aus vorgefertigten Komponenten aufgebaut. Die Effizienz des Konstruktionsprozesses sowie dessen Effektivität im Hinblick auf die später durchzuführenden Versuche sind stark von der Erfahrung des Entwicklungsingenieurs abhängig. Je größer dessen Know-How und dessen Überblick über frühere Versuche und deren Resultate, desto weniger Aufwand muß in die Erstellung einer prinzipiellen Lösung investiert werden. Desweiteren kann der Ingenieur die Gefahr von Fehlentscheidungen dadurch gering halten, daß er auf die Erfahrungen aus Projekten mit ähnlichen Zielsetzungen zurückgreift. In der Praxis tritt nämlich oftmals der Fall auf, daß ein Projekt durchzuführen ist, das sich aufgrund fast identischer Kundenanforderungen von früheren Projekten kaum unterscheidet, so daß bereits als geeignet erkannte Baukomponenten so weit als möglich wieder berücksichtigt werden können.

Der lückenlose Zugriff auf Informationen früherer Versuche ist demnach eine wichtige Aufgabe bei der Abwicklung eines neuen Versuchs, aber eben diese Aufgabe gestaltet sich in der momentanen Situation im Bereich Versuch extrem schwierig, da keine Möglichkeit vorhanden sind, solche Informationen zentral zu verwalten und abzufragen. Zwar existieren zu jedem früheren Versuch entsprechende Dokumentationen (Versuchsberichte, Versuchsergebnisse, etc.), nur werden diese in Ordnern gesammelt oder als Textdokumente abgelegt und sind somit für eine schnelle und flexible Informationsgewinnung ungeeignet. Außerdem sind diese Berichte oftmals nur den unmittelbar beteiligten Personen im Detail bekannt. Durch Beseitigung dieses Mangels könnten die Lösungsfindung verbessert, Fehler vermieden und Kosten eingespart werden.

Genau hier liegt einer der Anwendungsschwerpunkte von RITA. In Abschnitt 3 werden wir anhand der Daten und Tätigkeiten des vorliegenden Schrittes skizzieren, wie RITA bei der Konstruktion von Prototypen eingesetzt werden kann.

2.3 Ermitteln des Versuchsumfangs

Um die Tauglichkeit der im vorigen Arbeitsgang erstellten Prototyps zu testen, sind vielfältige Prüfungen notwendig. Dabei spielen Umfang und Reihenfolge der Prüfungen eine wichtige Rolle und müssen genauestens geplant werden, um zum einen die Kosten für die Tests und zum anderen die dafür benötigte Zeit zu optimieren. Für jede Prüfung wird eine Prüfvorschrift erstellt, die beschreibt,

- wie die Versuchsparameter definiert sind,
- mit welchen Bauteilen oder Baugruppen der Versuch durchgeführt werden soll,
- wie der Versuchsaufbau gestaltet wird,
- und welche Ergebnisse zulässig sind.

Zur Erstellung der Prüfvorschriften existieren Formulare, in denen für jeden Versuch die entsprechend gültigen Einträge vorgenommen werden. Der grundsätzliche Aufbau solcher Vorlagen ist gleich, jedoch können die Einträge durchaus von Versuch zu Versuch variieren. Daher hat die bisher übliche Vorgehensweise, aus Zeitersparnis ein altes Vorschriftendokument für den aktuellen Versuch zu kopieren, unter Umständen schwerwiegende Konsequenzen. So kann es passieren, daß ein Versuch mit falschen bzw. nicht aktuellen Vorschriften und somit mit inkorrekten Parametern durchgeführt wird. Es werden also falsche Ergebnisse geliefert, der Versuch ist unbrauchbar und muß wiederholt werden.

Auch das Ermitteln des Versuchsumfangs stellt einen wichtigen Anwendungsbereich für ein rechnergestütztes Informationssystem dar. So könnte RITA z. B. weiterhin die Übernahme früherer Prüfberichte unterstützen, allerdings in Verbindung mit Aktualitäts- und Korrektheitstests, die es erlauben, die zuvor genannten Fehlerquellen auszuschalten.

2.4 Kalkulation

Alle durchzuführenden Prüfungen werden in einer Kalkulationsmatrix erfaßt. Für die Kalkulation werden nun die Kosten und Zeiten entsprechend eingetragen. Um dem Entwicklungsrisiko Rechnung zu tragen, wird weiterhin die Anzahl der durchzuführenden Testläufe hinzugefügt. Ein zu prüfendes Bauteil, über das noch keine Erfahrungswerte vorliegen, bedeutet dabei ein höheres Risiko und macht somit eine größere Anzahl von Versuchen erforderlich. Aufgrund dieser Daten kann der entsprechende Zeit- und Kostenaufwand ermittelt werden. Die Kalkulationsmatrix enthält außerdem Einträge darüber, welche Bauteilmuster zur Abwicklung eines Versuches benötigt werden. Diese Aufstellung wird dann noch um die Baustände ergänzt, mit denen die Prüfung abgewickelt werden soll.

2.5 Aufstellen der Versuchsfolge

Den Abschluß der Versuchsvorbereitung bildet die Aufstellung einer Versuchsreihenfolge, die einen optimalen Ablauf bezüglich Zeitbedarf und Kosten gewährleistet. Dafür werden durch den Ingenieur Optimierungen durchgeführt. So bedeutet beispielsweise Kostenminimierung, nichtzerstörende vor zerstörenden Versuchen durchzuführen, und Zeitminimierung, die Prüfungen bezüglich Dauer und Ressourcenbedarf aufeinander abzustimmen. Andererseits gibt es in der Regel aber auch Prüfungen, die völlig unabhängig voneinander abgewickelt werden können, so daß hier keine feste Reihenfolge notwendig ist. Oftmals ist sogar das genaue Gegenteil der Fall - um Verzögerungen bei der Durchführung zeitlch vorgeplanter Tests zu überbrükken, ist es wünschenswert, solche unkorrelierten Versuche im zeitlichen Rahmen der gesamten Versuchsbearbeitung beliebig verschieben zu können.

Für ein Informationssystem bedeutet das, daß eine flexible, jedoch auf Vollständigkeit ausgerichtete Planung der Aktivitäten im Rahmen der Versuchsdurchführung unterstützt werden muß. Ein geeigneter Planungsmechanismus sollte es erlauben, neben fixen Ablaufreihenfolgen für die zu koordinierenden Aktivitäten auch zeitlich nicht aufeinander abgestimmte Versuche zu beschreiben und deren Ausführung zu überwachen.

2.6 Durchführung der Versuche

Neben der Abteilung Versuch selbst sind i. allg. weitere Firmenbereiche an der Durchführung von Tests beteiligt. So wird z. B. die Mustererstellung von einer eigenen Werkstatt erledigt, und Crashtests werden zentral an einem speziellen Prüfstand ausgeführt. Desweiteren gibt es Tests, die ohne Absprache mit anderen Abteilungen ablaufen können, so daß deren zeitliche Planung flexibler gestaltet werden kann.

Während im zuvor dargestellten Schritt die Planung von Arbeitsabläufen im Mittelpunkt stand, ist in der vorliegenden Phase eines Projekts deren Überwachung und Koordination von besonderer Bedeutung.

2.7 Erstellung des Versuchsberichts

Sind alle Tests durchgeführt und ausgewertet, müssen die Ergebnisse schriftlich festgehalten werden. Dies sollte in einheitlicher Form geschehen, da nur so der spätere systematische Zugriff auf die Daten ermöglicht wird. In diesem Zusammenhang sollte RITA die Versuchsingenieure durch Vorgabe von Bildschirmmasken bzw. Zugriff auf frühere Prüfberichte unterstützen. Genau wie bei der Ermittlung des Versuchsumfangs muß das System aber auch hier darauf achten, daß keine bzw. keine falschen Daten früherer Projekte übernommen werden, Korrektheitsprüfungen bzw. Plausibilitätstests sollten also auch in dieser Phase funktionaler Bestandteil von RITA sein.

3. Erstellung des Anforderungskatalogs

Im folgenden soll anhand des Teilschritts "Erstellung des Anforderungskatalogs" beispielhaft diskutiert werden, welche Daten und Operationen im Bereich Versuch zu warten bzw. bereitzustellen sind. Die dabei verwendeten Beispiele beziehen sich auf den Anforderungskatalog für den Fahrersitz eines PKWs. Bei der Darstellung haben wir uns eng an den aus dem Bereich objektorientierter Datenmodelle bekannten Beschreibungsmitteln ausgerichtet [2]. Die Entscheidung für eine objektorientierte Darstellung wird im Verlauf dieses Abschnitts näher erläutert. Wie wir in Abschnitt 4 diskutieren werden, erlaubt die nachfolgend verwendete Repräsentationsform die Realisierung des Systems sowohl mittels objektorientierter als auch objekt-relationaler Datenbanksysteme.

3.1 Zentrale Datenstrukturen

Wie bereits in Abschnitt 2.1 erwähnt, setzen sich Anforderungen aus drei Komponenten zusammen - den Kundenanforderungen, den Gesetzesvorschriften und den firmeninternen Normen. Allen Anforderungen sind weitere Eigenschaften gemein, weshalb eine Klasse *Anforderungen* eingerichtet wird, die die gemeinsamen Attribute definiert und vererbt (vgl. Bild 1). Zu diesen Eigenschaften zählen beispielsweise die Bezeichnung des Projekts, dem die Anforderungen zugeordnet sind, oder kundenspezifische Daten. In den Subklassen *Kundenanforderungen*, *Gesetzesvorschriften* und *Firmenvorschriften* werden weitere, spezifischere Attribute festgelegt. Für jede dieser Anforderungsbereiche existieren Ausprägungen (d. h., Instanzen). Dies liegt zum einen darin begründet, daß es verschiedene Kunden, Gesetze oder firmeninterne Vorschriften gibt, zum anderen kommt hinzu, daß z. B. je Kunde verschiedene Versionen von Vorschriften verwaltet werden müssen, um nicht nur auf frühere Versuche, sondern auch auf deren Anforderungen zurückgreifen zu können.

Der Anforderungskatalog für ein konkretes Projekt wird als Instanz der Klasse *Anforderungskatalog* erzeugt. Er enthält in der Regel eine Menge von Kundenanforderungen, Gesetzesvorschriften und Firmenvorschriften. Die Zuordnung dieser Spezifikationen zu einem Anforderungskatalog läßt sich in natürlicher Weise durch das Abstraktionskonzept der Assoziation abbilden [5, 10]. Zu diesem Zweck vererbt die Klasse *Anforderungskatalog* Attribute an ihre Instanzen, um auf die entsprechenden Anforderungen verweisen zu können. In den Instanzen der Klasse werden diese Verweise dann auf die für das Projekt relevanten Instanzen der Klassen *Kundenanforderungen*, *Gesetzesvorschriften* und *Firmenvorschriften* gesetzt.

Jeder dieser Instanzen sind eine Menge konkreter Prüfvorschriften assoziativ zugeordnet (in Bild 1 durch Prüfung P_1 angedeutet). Bei solchen Prüfungen handelt es sich einerseits um gesetzlich vorgeschriebene Tests, andererseits beinhalten sie vom Kunden bzw. vom Unternehmen selbst geforderte Probeläufe. In Bild 2 ist diese Aufteilung veranschaulicht. In nachfolgender Tabelle ist exemplarisch aufgeführt, welche Attribute in der Klasse *Prüfungen* definiert sind und an deren Subklassen weitervererbt werden. Dabei wird deutlich, daß neben konventionellen Datentypen auch Texte oder Graphiken zu berücksichtigen sind.

Gesetzliche Prüfungen und sonstige Prüfungen können überlappen. In Bild 2 ist z. B. die "Gurtverankerungsprüfung ohne Kraft" sowohl Teil der gesetzlichen als auch der allgemeinen Prüfungen. Die bereits im Zusammenhang mit Anforderungskatalogen genannten,

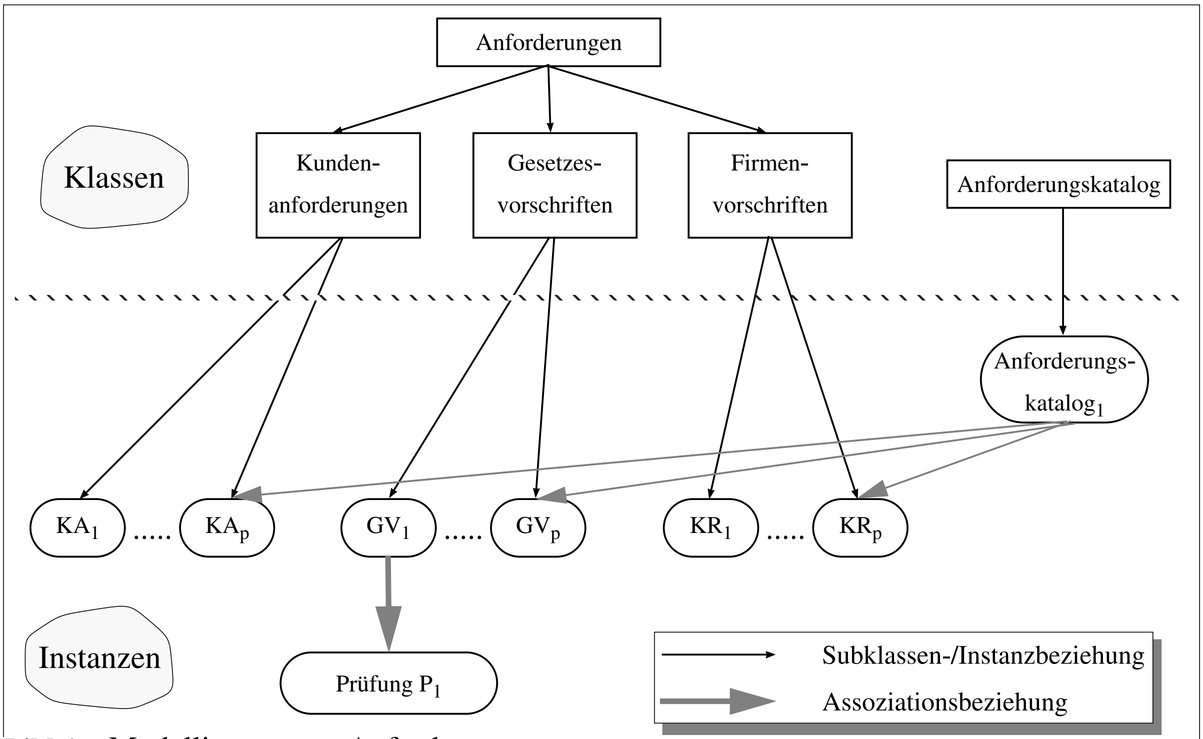

Bild 1: Modellierung von Anforderungen

konkreten Prüfvorschriften finden sich auch in der Hierarchie der Prüfungen wieder (vgl. Prüfung P_1 in Bild 2).

Weitere Aspekte der Modellierung in RITA betreffen beispielsweise die nähere Spezifikation der Eigenschaften der oben eingeführten Subklassen. Aus Gründen der Übersichtlichkeit werden diese hier nicht explizit aufgeführt, ebenso wie die operationalen Eigenschaften, die in den Klassen definiert werden.

Aus den zuvor angestellten Überlegungen wird deutlich, daß ein wesentlicher Vorteil der objektorientierten Repräsentation in der gegenüber herkömmlichen Datenmodellen reichhaltigen Palette an organisationalen Konzepten zur Strukturierung von Informationen liegt. Neben den bereits erwähnten Abstraktionskonzepten Generalisierung/Klassifikation und Assoziation ist auch das Konzept der Aggregation im Rahmen technischer Anwendungen unabdingbar [8]. Es ist zum Beispiel notwendig, um Sitze und deren Bauteile angepaßt darstellen zu können. Dies gilt nicht nur für die prototypischen Produkte im Bereich Versuch, sondern auch für die Phasen der Konstruktion und der Produktion. Auch ließe sich die Zuordnung von Anforderungen zu einem Anforderungskatalog durch die Aggregation anstelle der oben verwendeten Assoziation abbilden.

Neben der reinen Strukturierung von Daten bieten die Abstraktionskonzepte darüberhinaus auch bei der Verarbeitung solcher Informationen erhebliche Vorteile. Automatische Schlußfolgerungsmechanismen [10] garantieren den Erhalt modellinhärenter Konsistenzbedingungen, außerdem erleichtern die Abstraktionskonzepte die zusammenhängende Verarbeitung von z. B. Mengen oder Aggregaten.

Nachdem wir die Modellierung der wichtigsten Daten im Zusammenhang mit der Erstellung eines Anforderungskatalogs vorgestellt

Attribut	Typ	Kommetar
Prüfteil	<prueftnr>	spezieller Code
Prüfname	STRING	-
Prüfablauf	TEXT	Vorschrift, nach der die Prüfung durchzuführen ist
Prüfmuster	set of '(' <prueftnr> <anz> ')'	Aufzählung der verwendeten Prüfteile und deren Anzahl
Prüfaufbau	TEXT	Beschreibung des Versuchsstands, etc.
Prüfungsnummer	<pruefnr>	spezieller Code
Hinweise	TEXT	-
Skizze	BITMAP	z. B. Zeichnung des Versuchsaufbaus

haben, wollen wir anhand einiger Beispiele zeigen, wie diese Daten im praktischen Einsatz verwendet werden.

3.2 Beispielanfragen

Es gibt mehrere Achsen, entlang derer Anfragen an das zu entwickelnde Informationssystem gestellt werden können. Bezogen auf den Teilschritt "Erstellung des Anforderungskatalogs" sind dies die Projekte und die Prüfungen. Folgende Anfragen sind typische Beispiele dafür, wie RITA als Informationsquelle genutzt werden könnte, wenn bei der Erfassung eines neuen Anforderungskatalogs auf die Kataloge früherer Projekte zurückgegriffen werden soll:

"Finde alle Kundenanforderungen eines bestimmten Projekts."

Diese Anfrage kann sich sowohl auf die Informationen der Instanzen der Klasse *Kundenanforderungen* selbst beziehen, als auch auf die mit ihnen assoziierten Prüfungen.

"Finde alle Prüfvorschriften einer bestimmten Prüfungskategorie."

Zu den Prüfungskategorien zählen sowohl die gesetzlich vorgeschriebenen Prüfungen als auch die allgemeinen Prüfungen.

"Finde alle Prüfungen, die ein bestimmtes Prüfteil nutzen."

Um diese Anfrage auf der zuvor dargestellten Modellierung zu beantworten, müßten alle Instanzen der gesetzlichen und allgemeinen Prüfungen sequentiell durchsucht werden, was bei großen Datenbeständen zu ineffizient sein kann. Diese Beobachtung zeigt, daß die zuvor beschriebenen Datenstrukturen bei weitem nicht alle sinnvollen Einsatzmöglichkeiten angemessen unterstützen und daher weiter ausgebaut werden müssen. Da solche Änderungen der Modellierung auch nachträglich, im laufenden Betrieb des Systems nicht auszuschließen sind, müssen die Datenstrukturen auch dynamisch erweiterbar sein.

Nach diesem groben Überblick über Daten und Operationen, die von RITA unterstützt werden müssen, wenden wir uns nun einem weiteren konzeptionellen Schwerpunkt im Rahmen von RITA zu, der Beschreibung der Arbeitsabläufe eines Projekts.

4. Einbettung in eine Systemumgebung

Um besser abschätzen zu können, welche Software- bzw. Hardwareplattform für RITA am besten geeignet ist, tragen wir die Anforderungen zusammen, die ein solches Informationssystem in Bezug auf semantische Konzepte, Datenhaltung und Verarbeitungsumgebung stellt.

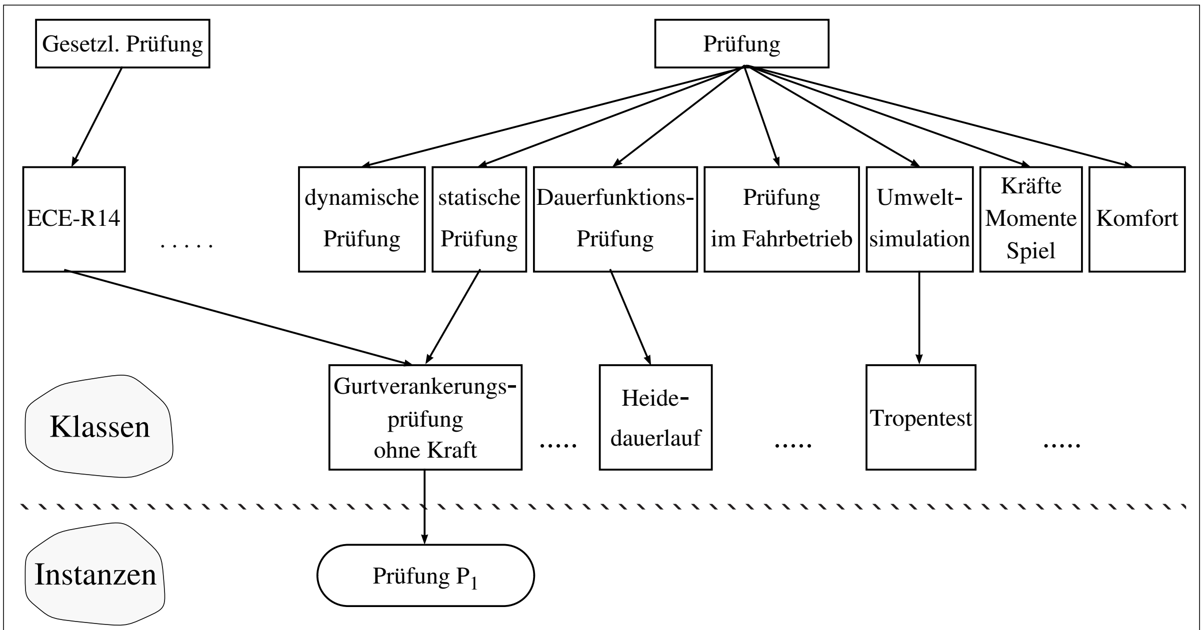

Bild 2: Modellierung von Prüfungen

4.1 Modellierung

Die im Rahmen von RITA zu bearbeitenden Datenmengen weisen folgende strukturellen Merkmale auf:

- stark strukturierte Datenmengen, wie sie als CAD-Objekte (z. B. technische Zeichnungen) in herkömmlichen Entwurfsanwendungen auftreten;
- große, einfach strukturierte Datenmengen, z. B. Kataloge für Bauteile oder Normen;
- neue Datentypen wie Bild oder Video, beispielsweise zur Darstellung von Details aus Versuchen.

Diese Anforderungen führen auf eine Vielfalt von Objekten und Beziehungen. Um diese angemessen modellieren zu können, sind zusätzlich folgende Anwendungscharakteristika zu berücksichtigen:

- potentiell große Mengen an Daten

Bereits aus der relativ knappen Darstellung eines einzelnen Teilschritts im Bereich Versuch dürfte deutlich werden, daß für ein Projekt umfangreiche Daten zu verwalten sind. Diese sind zum Teil für verschiedene Projekte identisch (z. B. Gesetzesvorschriften). Daher erscheint ein Datenbanksystem als Grundlage von RITA unbedingt notwendig, um effiziente Datenhaltung, Mehrbenutzerbetrieb und Sicherungsmaßnahmen (Recovery) garantieren zu können. Auch die von Datenbanksystemen angebotenen Anfragemöglichkeiten sind für RITA von großer Bedeutung, da Anfragen über den Daten einen zentralen Teil der Funktionalität von RITA ausmachen werden.

- Vielfalt verschiedener Datentypen

Neben Standard-Datentypen wie INTEGER oder STRING sind auch Texte und Bilder zu speichern und zu verarbeiten. Letzteres erfordert auch die Einbindung geeigneter Zugriffsfunktionen in die Modellierung. Fast alle kommerziellen relationalen, objektorientierten und objekt-relationalen Datenbanksysteme bieten solche Datentypen an.

- Daten sind teils hierarchischer Natur, teils "flach" strukturiert

Die in diesem Aufsatz beschriebenen Daten sind fast ausschließlich hierarchisch gegliedert. In den im vorigen Abschnitt gezeigten Beispielen kommen neben den Abstraktionskonzepten Generalisierung und Assoziation (in Form flacher Mengen) auch anwendungsspezifische Strukturierungen der Daten zum Einsatz. Diese Konzepte werden von objektorientierten und objekt-relationalen Datenbanksystemen unterstützt. Andere, hier nicht angesprochene Daten, wie zum Beispiel Bauteilkataloge, sind jedoch flach strukturiert.

- Aspekte der DB-Modellierung

Die skizzierten datenbezogenen Anforderungen können durch relationale Datenmodelle in ihrer momentanen Ausprägung und Standardisierung als SQL2 [4] nicht erfüllt werden. Wie bereits gezeigt, besitzen objektorientierte Datenmodelle einen reichhaltigen Vorrat an Datentypen und Strukturierungskonzepten, mit denen neben benutzerdefinierten Beziehungen auch Generalisierungs-/Spezialisierungshierarchien (als systemkontrollierte Beziehungen) aufgebaut werden können. Die zugehörigen DB-Sprachen sind typischerweise prozedural und satzorientiert; nur in Ausnahmefällen werden mengenorientierte Anfragesprachen (wie z. B. OQL) zur Verfügung gestellt. Eine ausschließlich navigierende Verarbeitung ist aus der Sicht der relationalen DB-Technologie, mit mengenorientierter Anfrageverarbeitung als dem Stand der Technik bei Datenbanksystemen, ein deutlicher Rückschritt, insbesondere weil weder Anfrageoptimierung noch Parallelisierung der Verarbeitung genutzt werden können. Andererseits sollten aber im Vergleich zu den relationalen Datenmodelllen komplexe Datentypen verfügbar sein. Deshalb gibt es seit kurzem Vorschläge für sogenannte objekt-relationale Modelle und Systeme. Dabei werden immer mindestens vier fundamentale Erweiterungen des relationalen Modells gefordert: Relationen als gültige Datentypen; mengenwertige Attribute; Prozeduren, die an Relationen gebunden sind; Vererbungshierarchien für Relationen. Diese Erweiterungen werden durch UniSQL verkörpert [6, 9]. In einem anderen Vorschlag werden als komplexe Datentypen mehr multimediale Datentypen wie Bild oder Sprache gefordert. Benutzerdefinierte Funktionen und benutzerdefinierte Operatoren, wie sie auch für SQL3 geplant sind, sollen die Handhabung dieser Datentypen in deskriptiven DB-Sprachen ermöglichen (z. B. in Illustra [14]).

Aus diesen Gründen scheint weder eine rein objektorientierte noch eine rein relationale Modellierung sinnvoll. Objekt-relationale Datenbanksysteme versprechen hier, durch ihre Ausdrucksmächtigkeit und Funktionalität, den idealen Mittelweg zwischen den genannten beiden Datenmodellen zu bilden. Vorschläge für objekt-relationale Datenmodelle erfordern momentan zwar noch erhebliche Forschungsarbeit, es werden jedoch schon "Start-up-Companies" gegründet, die entsprechende Datenbanksysteme entwickeln. Im Bereich objekt-relationaler Datenbanksysteme existieren derzeit bereits zwei Anbieter, Illustra [11] und UniSQL [9]. Beide bieten eine an SQL angelehnte Anfragesprache als Schnittstelle zum Benutzer.

Neben diesen rein datenbezogenen Anforderungen kommen noch weitere hinzu, die mit der Nutzung von heterogenen Datenbeständen oder Systemen, die ihre Daten in speziellen Formaten gekapselt halten (z. B. CAD-Systeme), zusammenhängen (siehe Abschnitt 4.3).

4.2 Workflow Management in RITA

Im vorigen Abschnitt haben wir die prinzipiellen Schritte im Rahmen eines Projekts beschrieben. Bild 3 faßt diese noch einmal in übersichtlicher Form zusammen. Eine Präzedenzstruktur bei der Bearbeitung besteht nur zwischen einigen dieser Schritte. Ebenso ist innerhalb einzelner Schritte ein paralleles

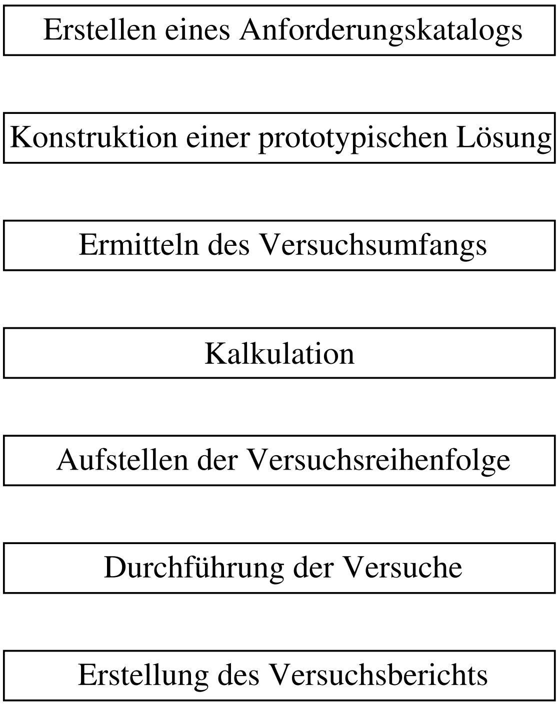

Bild 3: Die Arbeitsschritte eines Projekts

Vorgehen denkbar. Wir wollen nun diskutieren, welche Anforderungen an eine geeignete Ablaufsteuerung [13] sich daraus ergeben.

4.2.1 Charakteristika technischer Anwendungen

Ausgangspunkt für unsere Überlegungen sind die spezifischen Eigenschaften der im Rahmen eines Projekts anfallenden Einzelaktivitäten, da diese die Aufgaben, Möglichkeiten und Grenzen der Steuerung des Workflow bestimmen. Die Tätigkeiten lassen sich anhand dreier Kriterien klassifizieren:

- Planbarkeit

Das Kriterium *Planbarkeit* beschreibt die Möglichkeiten, einzelne Aktivitäten zu planen, indem für sie eine feste zeitliche Abfolge der Teilaktivitäten spezifiziert wird. Dabei lassen sich *geregelte* und *ungeregelte* Aktivitäten unterscheiden. Geregelte Aktivitäten sind solche, deren Ablauf exakt definiert werden kann, ungeregelte Aktivitäten lassen sich nicht in ein solches fixes Schema pressen. Bezogen auf den hier vorliegenden Anwendungsbereich ist zum Beispiel die Durchführung eines Versuchs eine geregelte Aktivität. Da Versuche aus Kostengründen so weit wie möglich an ein und demselben Prüfmuster ausgeführt werden sollen, muß garantiert sein, daß nicht zerstörende vor zerstörenden Versuchen abgewickelt werden. Auf der anderen Seite gibt es aber auch eine Reihe ungeregelter Aktivitäten, beispielsweise die Erstellung von Prüfberichten, für die zwar Vollständigkeit garantiert werden muß, ansonsten aber keine Angaben über die Vorgehensweise sinnvoll erscheinen.

- Gruppendynamik

Dieses Kriterium bezieht sich auf Aktivitäten, die in hohem Maße ungeregelt sind und die zu ihrer Durchführung vorläufige Informationen austauschen müssen (die ggf. zurückgezogen werden können). Weiterhin beschreibt dieses Kriterium, ob solche Aktivitäten *zentral*, unter Umständen von einer einzelnen Person, oder *verteilt*, unter Einbeziehung mehrerer Beteiligter, bearbeitet werden können. Im letzteren Fall muß angesichts der vorausgesetzten Ungeregeltheit von hoher Gruppendynamik und einem damit verbundenen hohen Kooperationsbedarf ausgegangen werden. Solcher Kooperationsbedarf kann zum einen aufgrund der Notwendigkeit entstehen, Informationen zwischen einzelnen Entscheidungsträgern auszutauschen. Dies ist zum Beispiel bei der Terminplanung für einen Versuch oder bei der Bewertung von Versuchsergebnissen der Fall. Ein weiterer Grund für Kooperation ist die Verteilung einer zusammenhängenden Aufgabe auf mehrere Personen. Hier läßt sich die Konstruktion einer prototypischen Lösung anführen, die in der Regel von mehreren Experten gemeinsam vorgenommen wird. Da es sich hier im wesentlichen um dynamische Entscheidungen handelt, sind geeignete Mechanismen zur Spezifikation und Erhaltung korrekter, kooperativer Abläufe zu entwickeln.

- zeitlicher Ablauf

Mittels dieses Kriteriums lassen sich Aktivitäten unterscheiden, die zeitlich voneinander abhängen, also *synchron* auszuführen sind, und solche, die in keinem zeitlichen Zusammenhang stehen (*asynchrone* Tätigkeiten). Als Beispiel für synchrone Tätigkeiten können der Bau von Prüfmustern und der Aufbau einer Versuchsanordnung genannt werden, da diese in engem zeitlichen Zusammenhang stehen. Asynchrone Aktivitäten sind u. a. isolierte Tests an unabhängigen Baugruppen eines Produktes.

4.2.2 Realisierungsaspekte

Aufbauend auf diesen Überlegungen ergeben sich eine Reihe von Anforderungen an ein für die vorliegende Anwendungsumgebung geeignetes Workflow Management. Als generelle Aufgabe soll es die Spezifikation von Daten- und Kontrollflüssen zwischen den zusammengehörigen Aktivitäten gestatten und zumindest selektiv transaktionale Eigenschaften - Datenintegrität sowie Korrektheit und Vollständigkeit von Programmausführungen - gewährleisten. Dabei ist zu beachten, daß die Aktivitäten eines Workflow lang andauernd (z. B. 10^7 sec und mehr) sind, was die Ansprüche an die Systemkontrolle und Verarbeitungskompatibilität (bei Systemevolution) erheblich verschärft [3]. Im einzelnen sind folgende Aspekte zu unterstützen:

Flexible Spezifikation des Ablaufs

Es muß möglich sein, einzelne Aktivitäten (insbes. die geregelten) inhaltlich und zeitlich zu beschreiben, sowie Kontrollstrukturen für deren Abwicklung zu definieren. Die Ablaufkontrolle muß dabei für die Dauer des gesamten Projekts, die in der Regel zwischen einigen Monaten und mehreren Jahren liegt, spezifizierbar sein und später auch gewährleistet werden.

Persistenz des Workflows

Zustandsinformation für Workflows muß in der betrachteten Systemumgebung persistent gespeichert sein, um Workflows zu vorgegebenen Zeitpunkten oder nach einer Systemunterbrechung durch einen Fehler (z. B. Crash) identifizieren und ohne Verlust oder Wiederholung von Arbeit fortsetzen zu können. Aus praktischen Erwägungen (nicht nur aufgrund der Dauer eines Projekts) sollte es außerdem möglich sein, den Arbeitsablauf gezielt zu unterbrechen und zu einem späteren Zeitpunkt wieder aufzunehmen. Dies setzt voraus, daß das Workflow Management alle zustandsbeschreibenden Informationen dauerhaft speichern und zugreifen kann.

Reaktivierung des Ablaufs

Bei Aktivierung des Ablaufs muß das Workflow Management den Benutzer zunächst mit den im momentanen Zustand relevanten Informationen versorgen. In erster Linie betrifft das die als nächstes auszuführenden bzw. ausführbaren Aktivitäten. Im Falle ungeregelter Tätigkeiten sollte das System ein Menü mit allen zulässigen Folgeschritten anbieten, aus dem dann die nächste Aktivität ausgewählt werden kann. Bei geregelten Aktivitäten sollten genaue Anweisungen angeboten werden, die das weitere Vorgehen exakt beschreiben.

Ganz gleich, ob es sich um ungeregelte oder geregelte Aufgaben handelt, sollte das System aber ggf. auch weitergehende Hinweise zur Durchführung des nächsten Schritts, wie z. B. Verweise auf evtl. vorhandene Informationsquellen, geben.

Korrektheitskontrolle

Neben der Kontrolle des Ablaufs selbst sollte es auch möglich sein, Korrektheitskriterien für einzelne Schritte zu spezifizieren, anhand derer dann überprüft werden kann, ob ein Schritt vollständig und korrekt abgearbeitet wurde. Während dies für einige Aufgaben teilweise automatisch geschehen kann (z. B. das Ausfüllen von Prüfberichten), muß dies bei anderen Tätigkeiten von Personen ausgeführt werden. Trotzdem sind aber auch in diesem Fall Rückmeldungen an das Workflow Management wichtig, da zum einen der weitere Verlauf eines Projekts davon betroffen sein kann, zum anderen Entscheidungen besser nachvollziehbar sind (was z. B. bei der späteren Wiederverwendung eines Prüfberichts sehr wichtig sein kann).

Kooperative Arbeitsumgebung

An einem Projekt sind viele verschiedene Personen beteiligt, die zum Teil parallel an gewissen Teilschritten arbeiten. Demzufolge muß eine geeignete Ablaufsteuerung Mechanismen bereitstellen, über die diese Gruppenaktivitäten koordiniert und kontrolliert werden können.

DB-bezogene Transaktionskonzepte sind zur Unterstützung kooperativer Arbeitsumgebungen derart zu erweitern, daß in Bearbeitung befindliche Informationen, d. h. unvollständige und nur graduell konsistente Dokumente, in flexibler Weise für Gruppenmitglieder verfügbar gemacht werden können.

Verknüpfung heterogener Komponenten

Zur Abwicklung eines Projekts sind eine Vielzahl rechnergestützter Arbeitsumgebungen und Informationsquellen vonnöten, z. B. CAD-Workstations, Meßstationen, Produktdatenbanken oder Textverarbeitungssysteme. All diese Komponenten müssen vom zu erstellenden Workflow Management nicht nur in einer uniformen Art und Weise zugreifbar gemacht werden, auch der Datenaustausch zwischen den Systemen muß in geregelter Form, anhand fest definierter Protokolle, übernommen werden.

Zusammenfassend ergeben sich als wichtigste Anforderungen aus unserer Diskussion folgende Realisierungsaspekte:

- Die Abläufe dauern sehr lange
- In der Regel ist nur eine Grobplanung der Abläufe möglich; nur bestimmte Anteile an der gesamten Arbeit sind geregelt, die anderen Anteile sind "frei" und werden durch die kreativen und intuitiven Entwurfsentscheidungen der Ingenieure bestimmt.
- Die Entwurfsarbeit erfordert Interaktion in der Gruppe mit zahlreichen Abhängigkeiten, die sich insbesondere durch die Bearbeitung gemeinsamer Daten ergeben.
- Eine systemgestützte Terminkontrolle der Arbeiten ist erforderlich, da diese langfristig sind und verteilt durchgeführt werden.
- Es soll personenunabhängig eine Integration des Firmenwissens und der vorhandenen Erfahrung in den Entwurfsablauf erreicht werden, was bei entsprechenden Arbeiten und Entscheidungen "aktive" Anstöße und Hinweise durch das Workflow Management voraussetzt.
- Die Abläufe finden in verteilten und heterogenen Entwurfs- und Produktionsumgebungen statt.

Aus der vorhergehenden Diskussion wurde die Vielfalt der von RITA zu unterstützenden Aktivitäten deutlich. Die Aufgaben an ein geeignetes Workflow Management sind deshalb als hoch anzusehen und stellen eine Herausforderung für die zukünftige Arbeit im Rahmen unseres Kooperationsprojektes dar.

4.3 Systemintegration

Zum einen ist es wünschenswert, die Datenbestände eines Unternehmens soweit wie möglich zentral zu verwalten. Häufig existieren daher bereits zentrale Server, auf denen auch RITA effizient abgearbeitet werden kann. Andererseits bieten, zumindest im Bereich Versuch, PCs die gewohnten Arbeitsumgebungen, die auch weiterhin beibehalten werden sollen. Ein weiterer wichtiger Punkt ist die Anbindung von RITA an bereits bestehende Software, z. B. CAD-Systeme oder andere, im Einsatz befindliche Datenbanken. Außerdem ist auf allen Ebenen (Rechner-Hardware, Peripherie, Software-Komponenten u. a.) mit häufigen Änderungen und schnellem Wachstum zu rechnen.

Bei solchen Anforderungen müssen rechnergestützte Informationssysteme vor allem wegen der Vielfalt der Hardware- und Software-Komponenten als offenes System, das einen hohen Grad an Heterogenität aufweist, konzipiert werden. Außerdem sind bereits existierende Hardware-/Software-Komponenten in das System zu integrieren. Wegen dieser Charakteristika bieten sich als Systemarchitekturen nur solche an, die auf dem Client/Server-Modell beruhen [1].

Das herkömmliche Client/Server-Modell basiert auf der kontextfreien Verarbeitung von Funktionsaufrufen. Im Rahmen von DB-Aufrufen und Transaktionskonzepten ist dieses abstrakte Verarbeitungsmodell jedoch viel zu einfach. Beispielsweise werden DB-Server in einer Transaktion mehrfach aufgerufen und müssen ihren Kontext bewahren. Außerdem sind referenzierte Daten für die Transaktionsdauer stabil zu halten und ggf. Änderungen (bei Rollback) zurückzusetzen. Bei Verteilung und bei heterogenen DB-Servern werden noch weitere Abhängigkeiten eingeführt. DB-Nutzung, Transaktionskonzept und Heterogenität erzwingen also eine beträchtliche Erweiterung des Client/Server-Modells [7].

Ein zweiter Problemaspekt betrifft das Verbergen der Heterogenität vor den Anwendungen, d. h. die Nutzung von standardisierten Kommunikations- und Kooperationsprotokollen für offene Systeme. Dabei sind vor allem das Zusammenspiel von Anwendung und DB-Server, die Kooperation heterogener Server (DB-Server) untereinander sowie die Integration von Workflow- und Transaktionskonzepten zu bewerkstelligen.

CORBA (Common Object Request Broker Architecture) scheint sich momentan als akzeptierter Standard für Client/Server-Umgebungen herauszubilden [12]. Als zentraler Bestandteil stellt der ORB (Object Request Broker) Funktionen für die Kommunikation in einer heterogenen Umgebung zur Verfügung. Fordert ein Client (Object) eine Dienstleistung an, so lokalisiert der ORB einen dienstbereiten Server und leitet den Aufruf an diesen weiter. Im Vergleich zum RPC bietet dieser Kommunikationsmechanismus deutlich mehr:

- Ortstransparenz, d. h., Clients sehen nur die ORB-Schnittstelle, die Auswahl des Servers erfolgt erst zur Laufzeit.
- Fehlertoleranz: Fällt ein Server aus, so kann der ORB nachfolgende Aufträge an einen anderen Server leiten oder einen neuen Server dafür erzeugen.
- Heterogenität: Der ORB-Mechanismus verbirgt neben der Verteilung die Heterogenität der Objekte.

Ein erweitertes Client/Server-Modell zusammen mit standardisierten Kommunikationsprotokollen sollte für rechnergestützte Informationssysteme das geeignete Architekturkonzept darstellen, das Kosteneffektivität, Offenheit und Flexibilität, was die Anpassung des Systems an wachsenden Leistungsbedarf und veränderte Systemumgebung (Skalierbarkeit) angeht, verkörpert. Jedoch sind momentan die speziellen Anforderungen einer objekt-relationalen, d. h. auch mengenorientierten Datenverwaltung sowie der Workflow- und Transaktionskonzepte und die Schwierigkeiten, sie mit einem Mechanismus wie CORBA zu integrieren, noch nicht geklärt.

5. Realisierung mit einem objekt-relationalen System

Zum Abschluß dieses Aufsatzes wollen wir noch kurz skizzieren, wie die zuvor beschriebenen Datenstrukturen mit Hilfe eines konkreten kommerziellen Systems, dem objekt-relationalen Datenbanksystem UniSQL [6, 9] realisiert wurden. Wir konzentrieren uns im folgenden auf die Klasse *Prüfung*, deren direkte Subklassen (vgl. Bild 2) sowie auf deren Beziehung zu *Anforderungen* (skizziert in Bild 1).

```
create class pruefung
    ( hinweise string,
      pruefablauf string,
      pruefaufbau string,
      pruefmuster set (string, integer),
      pruefname string,
      pruefteil set (string),
      pruefungsnummer string,
      ..........
      skizze x11bitmap);
```

Bild 4: Initiale Definition der Klasse *Prüfung*

Zunächst muß die Klasse *Prüfung* kreiert werden. Gemäß den vorhergehenden Überlegungen stehen ihre Instanzen mit Instanzen der Klasse *Anforderungen* in Beziehung, was durch ein mengenwertiges Attribut ausgedrückt werden kann, welches Verweise auf entsprechende Instanzen, d. h. die zugehörigen OIDs, enthält. Da die Klasse *Anforderungen* erst noch definiert werden muß, enthält die initiale Klassenspezifikation für *Prüfung* dieses Attribut aber noch nicht. Wie wir nachfolgend zeigen werden, wird die Klasse später erst um diese Eigenschaft erweitert. Bild 4 zeigt die anfängliche Definition der Klasse *Prüfung*. Gemäß Abschnitt 4.1 enthält diese Abbildung nur einige repräsentative Attribute, die für Prüfungen relevant sind. Das Attribut *pruefteil* enthält die Menge der Prüfteile, die für einen bestimmten Test benötigt werden. Da UniSQL benutzerdefinierte Datentypen nur unzulänglich unterstützt, mußten wir Bezeichner wie pruefname durch Strings repräsentieren. Zur Darstellung von Prüfskizzen benutzten wir den von UniSQL standardmäßig angebotenen Datentyp *x11bitmap*.

```
alter class pruefung
    add attribute element_of_anforderungen
      set (anforderungen) not null;
```

Bild 5: Erweiterung der Klasse *Prüfung*

Jetzt können die Klasse *Anforderungen* sowie deren Subklassen deklariert werden. Damit wird es nun möglich, das o.g. mengenwertige Attribut einzurichten, das die Beziehung zwischen Prüfungen und Anforderungen herstellt. Der entsprechende Befehl ist in Bild 5 dargestellt. Da eine Prüfung immer mindestens einer Anforderung zugeordnet ist, haben wir der Definition als Integritätsbedingung hinzugefügt, daß das Attribut keine leere Menge enthalten darf. Genauere Festlegungen von Unter- bzw. Obergrenzen werden von UniSQL nicht angeboten.

Insgesamt ergibt sich die in Bild 2 gezeigte Modellierung. In dieser Abbildung sind mengenwertige Attribute bzw. Attribute, die Verweise auf Objekte enthalten, noch einmal gesondert dargestellt. So läßt sich aus Bild 2 auch ersehen, daß der vordefinierte Datentyp *x11bitmap* in UniSQL durch eine eigene Klasse repräsentiert wird.

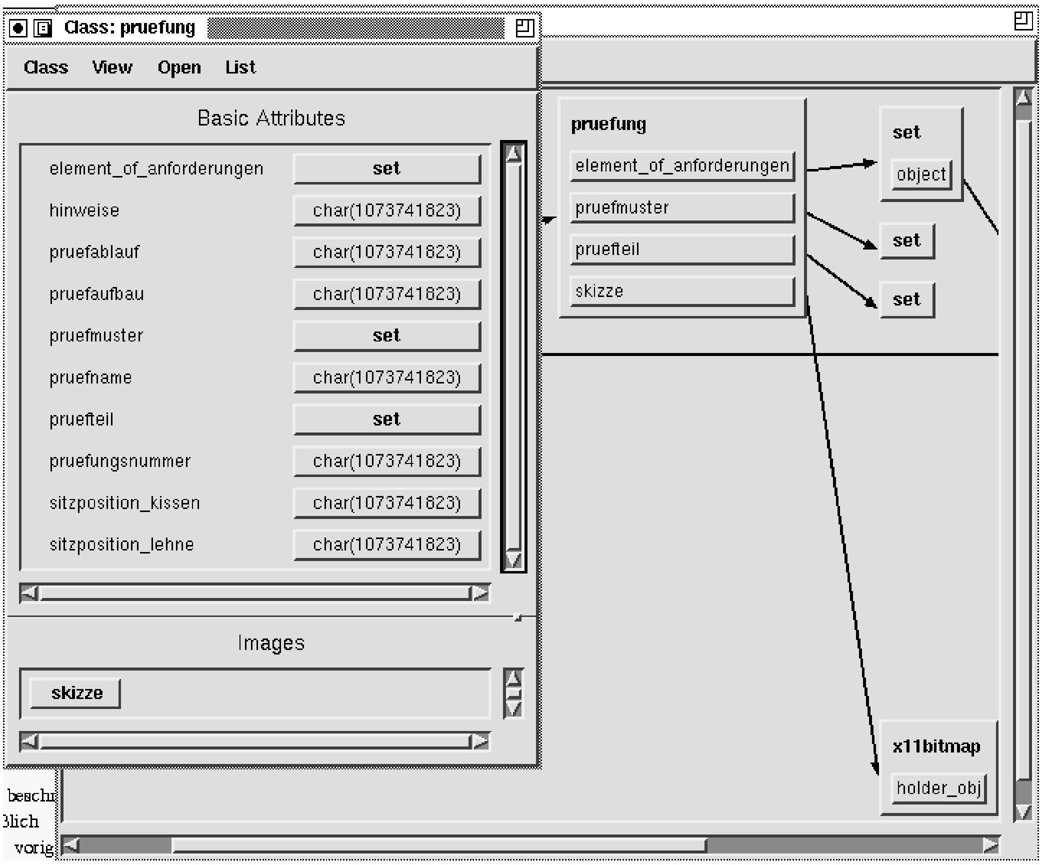

Bild 6: Repräsentation von Prüfungen als DB-Schema in UniSQL

Bild 6 zeigt darüberhinaus noch einmal sehr deutlich Schwächen des von UniSQL bereitgestellten objekt-relationalen Datenmodells auf. Sämtliche Beziehungen zwischen Objekten, die nicht durch die systemkontrollierte Vererbung modelliert werden können, müssen als benutzerdefinierte Beziehungen durch mengenwertige Attribute repräsentiert werden. Auf diese Weise werden sie systemseitig gleichförmig dargestellt und nur durch einfache Operationen unterstützt, was einer bloßen syntaktischen Verarbeitungskontrolle entspricht. In Bild 6 betrifft das die Attribute element_of_anforderungen, pruefmuster und pruefteil. Hier könnten verbessere Strukturierungskonzepte ihre durchaus unterschiedliche Semantik berücksichtigen.

Mit den fehlenden Abstraktionskonzepten von Assoziation und Aggregation geht einher, daß auch systemkontrollierte Operationen fehlen, wenn die Modellierung dieser Beziehungen mit Hilfe einfacher mengenwertiger Operationen erfolgt. Bei dieser Art der anwendungskontrollierten Simulation von Abstraktionsbeziehungen bleibt die Verantwortung für Leistung und Konsistenz beim Benutzer.

Die fehlende Möglichkeit, anwendungsspezifische Datentypen zu definieren, hat zur Folge, daß alle Datentypen, die über die standardmäßig angebotenenen hinausgehen, in Strings verpackt werden müssen. Dies ist nicht nur aus semantischen Aspekten unbefriedigend, auch aus Effizienzgründen ist diese Lösung nicht wünschenswert; man denke beispielsweise nur an das Attribut pruefungsnummer, dessen (in der Regel wohl sehr kurze) Werte durch Strings von maximal 1GByte dargestellt werden müssen.

6. Ausblick

Naturgemäß sollen Projekte der beschriebenen Art Meilensteine in der Entwicklung von Informationssystemen setzen. Die dabei vorherrschende wissenschaftliche Fragestellung verlangt die Vorgabe herausfordernder Ziele und die Nutzung innovativer Ansätze. Auf der anderen Seite hat sich eine praktische Lösung an vielfältige Gegebenheiten zu orientieren, die ihre Tauglichkeit für den betrieblichen Einsatz wesentlich bestimmen.

Aus Sicht der Forschung sollten alle Anforderungen der diskutierten Problemfelder durch ideal passende Konzepte und Modelle beschrieben und einer Systemlösung zugeführt werden. Die erwähnten Randbedingungen der Praxis erzwingen jedoch in vielen Fällen den Einsatz existierender Systeme, was eine Reduzierung der Anforderungen und der Modellierungsmöglichkeiten nach sich zieht. Dies darf jedoch nicht auf Kosten der Zuverlässigkeit und Leistung des zu realisierenden Systems gehen, d. h., die fehlende Funktionalität muß durch die Anwendung abgedeckt werden. Aus diesen Gründen kommen die aus unserer Sicht leistungsfähigsten Lösungen, wie z. B. KRISYS [10] für die Datenhaltung, ein angepaßtes Kommunikationsprotokoll sowie ein auf die spezielle Anwendung zugeschnittenes Workflow Management, nicht in Frage. Stattdessen werden mit UniSQL und CORBA kommerzielle Produkte herangezogen. Da es für Workflow Management momentan nur wenig geeignete Lösungen gibt, ist in diesem Bereich eine angepaßte Systementwicklung wohl unumgänglich.

Literatur

[1] Buck-Emden, R., Galimow, J.: Die Client/Server-Technologie des SAP-Systems R/3, Addison-Wesley, 2.Auflage, 1995.

[2] Cattell, R. (ed.): Next Generation Database Systems, Special issue of Comm. of the ACM, Vol. 34, No.10, 1991.

[3] Dayal, U., Shan, M.-C.: Issues in Operation Flow Management for Long-Running Activities, Data Engineering Bulletin, Vol. 16, No. 2, June 1995, pp. 41-44.

[4] Date, C.: An Introduction to Database Systems , Addison-Wesley, 1995.

[5] Deßloch, S., Härder, T., Mattos, N., Mitschang, B.: KRISYS: KBMS Support for Better CAD-Systems, Proc. Int. Conf. on Data and Knowledge Systems for Engineering and Manufacturung, Gaithersburg, MD, Oct. 1989, pp. 172-182.

[6] Gala, S., Kim, W.: Database Design Methodology: Converting a Relational Schema into an Object-Relational Schema, Proc. ACM Symp. on Advanced Database Systems and Their Integration, Japan, Oct. 1994.

[7] Gray, J., Reuter, A.: Transaction pro-cessing: concepts and techniques. Morgan Kaufmann Publ. San Mateo, CA. 1993

[8] Hübel, Ch., Sutter, B.: DB-Integration von Ingenieuranwendungen - Modelle, Werkzeuge, Kontrolle, Vieweg, 1993.

[9] Kim, W.: On Marrying Relations and Objects: Relation-Centric and Object-Centric Perspectives, Proc. 4th Int. Conf. on Database Systems for Advanced Applications'95, Singapore, 1995, pp. 31-37.

[10] Mattos, N.: An Approach to Knowledge Base Management, LNCS 513, Springer-Verlag, 1991.

[11] Olson. M.: Cover your assets, Industrial Session 7, 1995 ACM SIGMOD Conf., San Jose, USA.

[12] Orfali, R., Harkey, D., Edwards, J.: Essential Client/Server Survival Guide, Van Nostrand, New York, 1994.

[13] Reinwald, B.: Workflow-Management in verteilten Systemen, B.G. Teubner, 1993.

[14] Stonebraker, M.: Object-Relational DBMS - The Next Wave, Illustra Information Technologies, Inc., http://www.illustra.com, 1995.