Integration von Datenmodellen - Eine Technologie für Facility Management

Peter Kolbe, Stefan Pfennigschmidt, Peter Jan Pahl

Facility Management

Bauwerke werden heute mit umfassender Systematik geplant, genehmigt und ausgeführt. Die Sicherheit und die Brauchbarkeit der Bauwerke werden durch Normung und Prüfung intensiv beeinflußt. Der Computereinsatz ist in diesem Bereich allgemein üblich. Im Gegensatz zu Planungs- und Ausführungsphase eines Bauwerks ist die Betriebsphase wenig systematisiert. Dies ist überraschend, da die Betriebsphase über die Lebensdauer eines Bauwerks hinweg wesentlich größere Kosten verursacht, als der Rohbau und da der Nutzen des Bauwerks ja nicht in seiner Errichtung sondern in seinem Betrieb liegt. Die mangelnde Systematik der Bewirtschaftung ist beispielsweise daran ersichtlich, daß es im allgemeinen noch keine brauchbare Dokumentation der Gebäudebewirtschaftung, wenige Normen und wenig Software für diesen Aufgabenbereich gibt.

Die Bewirtschaftung eines Bauwerks ist ein komplexer technischer und organisatorischer Vorgang. Der Begriff Bewirtschaftung umfaßt die Belegung des Bauwerks, seinen Betrieb und seine Instandhaltung. Wirtschaftliche, technische und administrative Gesichtspunkte der Nutzung und des Werterhalts sind gleichermaßen von Bedeutung. Eine solche komplexe Aufgabe kann nur mit einer zuverlässigen Informationsbasis bewältigt werden, die in kurzen Zeiträumen aktualisiert wird und an vielen Orten einsetzbar ist. In der Informations- und Kommunikationstechnik wird die Bereitstellung und Nutzung einer solchen Informationsbasis Facility Management genannt.

Sowohl industrielle Unternehmen als auch öffentliche Institutionen betrachten Facility Management zunehmend als zentrale betriebswirtschaftliche Aufgabe. Charakteristisch für die baulichen Anlagen dieser Betreiber ist, daß sie räumlich verteilt sind, ihre Betreuung, Nutzung und Instandhaltung jedoch koordiniert sein muß und diese Aufgabe nur arbeitsteilig bewältigt werden kann. Durch Facility Management sollen die zufälligen Abläufe während der Nutzungsphase von Bauwerken durch gesteuerte Anpassungen an sich ändernde Anforderungen ersetzt werden. Das Ergebnis ist eine Optimierung der Bewirtschaftung.

Bei näherer Betrachtung der Art der Informationen, die durch ein Facility-Management-System verarbeitet werden, kristallisieren sich drei Schwerpunkte heraus: Informationen über den Gebäudebestand, Informationen über den Geschäftszweck und Informationen über die Betriebswirtschaft. Aufgrund der Individualität eines jeden Bauwerks, seiner Funktion und der Organisationsstruktur seiner Nutzer unterscheiden sich die Anforderungen für jedes zu installierende System. Daraus folgt, daß es nicht *das* Facility-Management-System geben kann. Des weiteren existieren häufig Anwendungsprogramme und entsprechende Datenbestände, die die Informationsbasis eines Facility-Management-Systems bilden. Beim Aufbau eines Facility-Management-Systems kann aus Zeit- und Kostengründen diese Funktionalität nicht stets neuentwickelt und die Datenbestände nicht neuerfaßt werden. Die Herausforderung besteht nun darin, ein System anzubieten, welches einfach an die individuellen Anforderungen angepaßt und bei Veränderungen umgestellt werden kann. Ein Facility-Management-System kann demzufolge nur als eine Kopplung von Bausteinen gesehen werden. Für diese Kopplung ist ein Integrationskonzept zu schaffen. Dabei ist sowohl der Datenaustausch zwischen den integrierten Bausteinen, als auch die übergreifende Auswertung des Gesamtdatenbestandes zu ermöglichen.

Eine grundlegende Funktionaliät von Facility-Management-Systemen ist die Dokumentation und Archivierung von Gebäudeinformationen. Die Verwendung standardisierter Methoden für Datenhaltung und -austausch ist deshalb essentiell. Das hier vorgestellte System verwendet Beschreibungs- und Implementierungsmethoden der STEP-Norm.

Zielstellung für die Integrationslösung

Die Anforderungen von Facility Management können nur durch ein verteiltes System befriedigt werden. Hierbei bearbeiten mehrere Anwender logisch einen Gesamtdatenbestand, der jedoch physisch verteilt ist. Durch die unterschiedlichen Teilaufgaben entstehen Datenbestände unterschiedlicher Struktur. Bei der Integration werden die einzelnen Datenbestände nicht verändert, so daß die Anwendungsprogramme, mit

denen die Daten erzeugt wurden, nach wie vor darauf zugreifen können. Somit bleibt die ursprüngliche Funktionalität erhalten.

Neben der Bildung eines logischen Gesamtdatenbestandes ist der Informationsaustausch zwischen den beteiligten Prozessen ohne deren Änderung zu realisieren. Durch das Integrationskonzept können Informationen der Teildatenbestände kontrolliert in das Gesamtsystem eingebracht werden. Das System ist so auszurichten, daß die Verfügbarkeit der Datenbestände flexibel ist sowie die Möglichkeit zur Eliminierung alter und die Einbindung neuer Bestände gewährleistet werden kann. Ziel ist die transparente Nutzung von Informationen verteilter Datenbestände, d.h. ohne Kenntnis der Lokalität der Datenmodelle.

Ein Laufzeitsystem für verteilte Datenbestände

Für eine Integrationslösung müssen Verbindungselemente zwischen den Teilprozessen eingeführt werden. Die prinzipiellen Möglichkeiten sind in Bild 1 dargestellt. Die erste Möglichkeit besteht darin, zwischen alle Prozesse Transformatoren zu schalten, so daß jeder Prozeß mit jedem verbunden ist. Diese Variante wird Punkt-zu-Punkt-Kopplung genannt. Sie bietet Vorteile, da die Transformatoren speziell auf die Eigenheiten der zu verbindenden Prozesse zugeschnitten werden können. Diese Lösung ist jedoch bei steigender Anzahl von Prozessoren zum Scheitern verurteilt, da dann die Anzahl der Transformatoren schnell anwächst. Betrachtet man die Anforderungen des Anwendungsgebietes Facility Management, so ist durch die Individualität und Veränderbarkeit der Datenmodelle eine Realisierung dieser Form nahezu ausgeschlossen.

Um dem überproportionalen Ansteigen der Transformatoranzahl bei der Punkt-zu-Punkt-Kopplung entgegenzuwirken, wird häufig eine Stern-Kopplung angestrebt. Hierbei ist für den Datenaustausch ein einheitliches Protokoll zu definieren; die Anzahl der Transformatoren reduziert sich dann auf die Anzahl der Prozesse. Das Problem besteht jedoch in der Definition des einheitlichen Protokolls. Verwendet man hierbei nur die Schnittmenge der Informationen aller beteiligten Prozesse, so ist bei der Übertragung mit hohen Verlusten zu rechnen. Wird jedoch die Obermenge der Informationen aller Prozesse im Protokoll aufgenommen, entstehen unübersichtliche, vielfach redundante und uneindeutige Protokolle.

Problematisch ist bei beiden Varianten das Einbringen neuer Informationen in den Gesamtdatenbestand, da

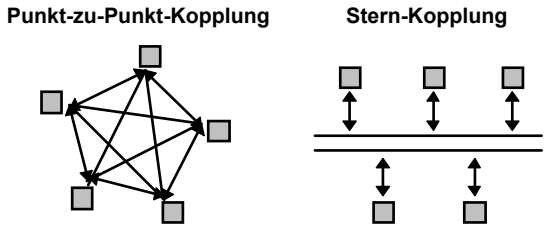

Bild 1: Möglichkeiten der Schnittstellengestaltung: spezifisch (links) oder standardisiert (rechts)

Transformatoren gewöhnlich einen Ausgangsdatenbestand in einen Zieldatenbestand überführen, ohne dabei zu prüfen, ob die Objekte im Zieldatenbestand eventuell bereits vorhanden sind. Die Änderung des Schemas eines einzigen Modells führt bei beiden Varianten zu umfangreichen Änderungen am Gesamtsystem, oder die neuen Informationen sind in keinem anderen Prozeß verfügbar.

Gesucht ist eine Lösung, die die Vorteile beider Varianten vereinigt und die Nachteile möglichst minimiert. D.h. die Flexibilität der spezifischen Schnittstellengestaltung sollte erhalten bleiben, der Programmieraufwand aber möglichst minimal gehalten werden. Als verbindendes Element wird nicht ein einheitliches Datenmodell angestrebt, sondern die Verwendung einer einheitlichen Zugriffsschicht bei freier Definierbarkeit der beteiligten Datenmodelle. Die Menge der Methoden dieser Zugriffsschicht wird Laufzeitsystem genannt. Das Laufzeitsystem gewährleistet den Datenzugriff, versteckt dabei jedoch die Implementierungsdetails der persistenten Datenhaltung und der Modellverteilung.

Erste Voraussetzung für das Funktionieren des Datenzugriffs ist die explizite Beschreibung der Datenstrukturen. Hierfür existieren Modellbeschreibungssprachen, wie beispielsweise die konzeptuelle Sprache EXPRESS. Die zweite Voraussetzung für den Datenzugriff ist die Existenz eines allgemeingültigen Funktionssatzes, der anhand der EXPRESS-Datenstruktur den Zugriff auf ein Modell, auf Instanzen des Modells und auf die Attribute der Instanzen ermöglicht. Die STEP-Norm schlägt hierzu die Funktionsbibliothek SDAI (*Standard Data Access Interface*) vor. Um nun Daten zwischen zwei Modellen, deren Schemata verschieden sind, auszutauschen, sind die Informationen zu erfassen, die die Überführungsfunktionen beschreiben. Die Zugriffsschicht wird um Algorithmen erweitert, die zur Laufzeit einen Verbund von Datenmodellen zur Verfügung stellt. Die detaillierte Beschreibung dieser Vorgehensweise ist Gegenstand des Beitrages „Integration von Datenmodellen - Eine Methodik für den Produktdatenaustausch“ in diesem Band.

Die Verwendung einer einzigen Funktionsbibliothek wie SDAI beschränkt die Integrationsmöglichkeit auf Datenbestände im STEP-21-Dateiformat. Um gleichzeitig auf Datenbestände verschiedener Datenhaltungssysteme wie Dateisysteme oder Datenbank-Management-Systeme zugreifen zu können, ist eine abstrakte Schnittstelle für den Datenzugriff zu spezifizieren und für jedes Datenhaltungssystem eine eigene Implementation anzubieten. Diese Möglichkeit ist Gegenstand des Beitrags „Ein Laufzeitsystem für dynamische Objektstrukturen für Entwurfsanwendungen“ in diesem Band.

Bild 2 zeigt die Architektur eines auf dem Laufzeitsystem aufbauenden Facility-Management-Systems. Ausgangspunkt sind Datenbestände unter-schiedlicher Teilbereiche wie Bauwerk, Personal, Produktion usw. Diese Datenbestände werden durch Fachingenieure mit fachspezifischen An-wendungsprogrammen bearbeitet. Die Verbindung dieser Datenbestände zu einem logischen Gesamtdatenbestand bietet dem Facilty Manager die Informationsbasis für Entscheidungen bezüglich der Gebäudebewirtschaftung. Der Verbund wird von einer Person - dem Integrator - vorgenommen, der mit Hilfe von Integrationswerkzeugen die Strukturen der Ausgangsdatenbestände analysiert und die Überführung zum logischen Gesamtdatenbestand konfiguriert. Dadurch schafft der Integrator eine Arbeitsumgebung für den Facility Manager.

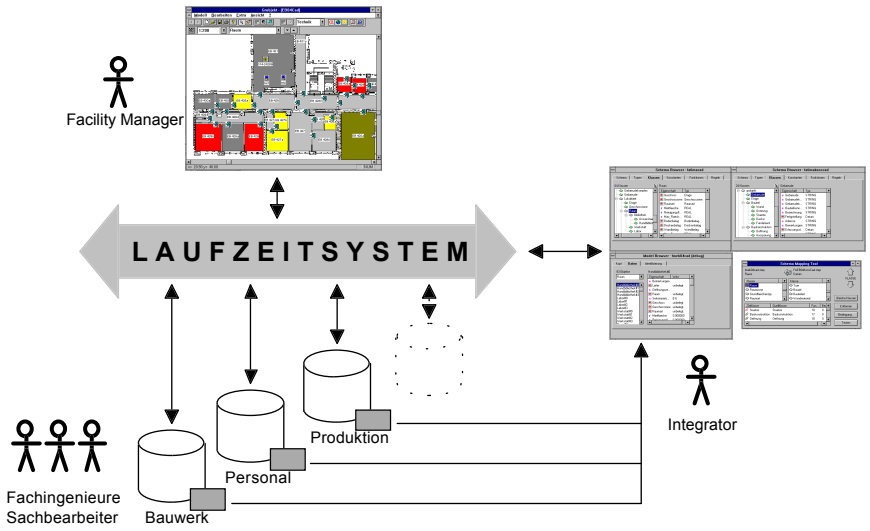

Bild 2: Das Laufzeitsystem als integrierende Komponente für ein Facility-Management-System

Integrationswerkzeuge

Bei der Erstellung des logischen Gesamtdatenbestandes müssen durch *Schema Mapping* die Strukturen der Datenbestände angepaßt werden und durch *Object Matching* die Zuordnung zwischen den Objekten hergestellt werden. Beim *Schema Mapping* sind beispielsweise folgende Konflikte zu lösen. Namenskonflikte treten auf, wenn äquivalente Klassen oder Attribute unterschiedlich bzw. verschiedene Klassen oder Attribute gleich benannt worden sind. Strukturkonflikte treten auf, wenn Informationen einerseits als Klassenattribute und andererseits als Klassenaggregation vorkommen. Oder wenn Informationen einerseits in einer feingegliederten Klassenstruktur abgelegt sind und andererseits diese Informationen in einem Attribut codiert sind. Weitere Konflikte treten auf, wenn Informationen in verschiedenen Maßeinheiten vorliegen oder in einem Attribut zusammengefaßt sind oder gar nicht exisitieren.

Nachdem die Strukturunterschiede überwunden sind, muß eine Zuordnung der Objekte getroffen werden. Das zu lösende Problem beim *Object Matching* besteht in der eventuell unterschiedlichen Identifizierung der Datenbestände. Teilweise sind die Objekte nicht oder nur implizit identifiziert, die Identifizierung ist nicht persistent - also dauerhaft über die Lebenszeit des Objekts, es wurden unterschiedliche Identifizierungsmechanismen verwendet oder die Datenbestände sind fehlerhaft, indem die Eindeutigkeit verletzt ist. Für den Modellverbund wird eine Auswahl von Attributen einer Klasse zur Identifizierung verwendet. Die Attribute können als Parameter von Funktionen angegeben werden, die dann bei der Suche nach dem Objekt gleicher Identifikation evaluiert werden. Dieser Mechanismus kann für beliebige Datenbestände angewendet werden.

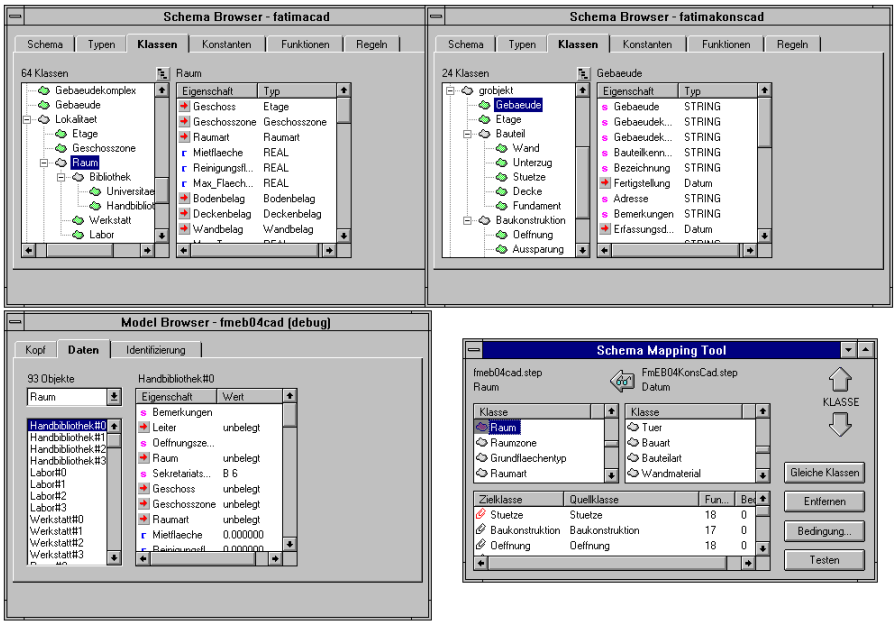

Bild 3: Schema-Mapping-Tool (rechts unten) in Kooperation mit je einem Schema-Browser für Quellschema (rechts oben) und Zielschema (links oben) und dem Model-Browser zur Darstellung des Modellverbunds

Für den logischen Gesamtdatenbestand muß genau ein Schema existieren, nach dem alle Objekte des Modellverbunds strukturiert sind. Dieses Zielschema und die Schemata der Quellen lassen sich durch einen *Schema-Browser* analysieren, um die Zuordnungen zwischen den Quellen und dem Ziel treffen zu können. Der *Schema-Browser* zeigt hierfür die Menge der Klassen, ihre Vererbungsbeziehungen sowie die Attributnamen und -typen jeder Klasse an. Ein Verfolgen der Klassenbeziehungen über relationale Attribute (Assoziationen) ist möglich. Die Semantik eines Schemas ist durch das parallele Browsen durch einen entsprechenden Datenbestand einfacher zu erfassen. Für diese Aufgabe wurde ein *Model-Browser* geschaffen. Er zeigt strukturorientiert die Objekte der einzelnen Klassen, sowie ihre Attributbelegungen an. Ein Verfolgen der Objektbe-ziehungen ist ebenfalls möglich. *Model-* und *Schema-Browser* können in Kooperation eingesetzt werden.

Für die Beschreibung der Beziehungen zwischen Quell- und Zieldatenbestand wird das *Schema-Mapping-Tool* eingesetzt (Bild 3). Es teilt sich in drei Ebenen: Modellebene, Klassenebene und Attributebene. In der ersten Ebene werden alle zu verbindenden Quelldatenmodelle erfaßt. Durch das Modell ist das Schema des Datenbestandes gegeben. In der zweiten Ebene ist die Zuordnung einander entsprechender Klassen möglich. Entsprechen mehrere Quellklassen nur einer Zielklasse, so sind entsprechend viele Klassenpaare anzugeben. Für den umgekehrten Fall, also die Zuordnung von einer Quellklasse auf mehrere Zielklassen, müssen zusätzliche Bedingungen angegeben werden. Existieren beispielsweise im Quellschema nur eine Klasse für Räume und dagegen im Zielschema Klassen für Büros, Flure und Bibliotheken, so kann über eine Bedingung (hier mittels Attribut Raumart der Klasse Räume) angegeben werden, zu welcher Klasse im Zieldatenbestand ein Objekt überführt werden soll. In der dritten Ebene des *Schema-Mapping-Tools* ist die Zuordnung der Attribute, sowie die Angabe der Identifizierungseigenschaften möglich. Im Normalfall werden die Attribute einfach einander zugewiesen. Reicht dies nicht aus, so können Attribute oder auch Konstanten als Parameter einer Funktion angegeben werden, die bei der Zuweisung an das Zielattribut ausgewertet wird. Sobald das Zielschema, ein Quelldatenbestand, mindestens eine Klassenbeziehung und einige Attributbeziehungen eingegeben sind, kann der Modellverbund getestet werden. Der logische Gesamtdatenbestand ist dann wahlweise im *Model-Browser* oder in einem generischen Anwendungsprogramm darstellbar. Somit kann der Integrator interaktiv den Modellverbund erstellen.

Generische Anwendungsprogramme

Um nicht jedes Facility-Management-System neuentwickeln und bei Bedarf umprogrammieren zu müssen, sollte Funktionalität möglichst allge-meingültig entwickelt und so konzipiert werden, das das zugrundeliegende Schema austauschbar ist. Beispiele für solche Funktionalität sind das Erzeugen, Bearbeiten und Löschen von Objekten beliebig definierbarer Objekttypen und die Erstellung einer grafischen Repräsentation für Objekte. Programme, die eine solche Funktionalität

bereitstellen und dabei schemaunabhängig arbeiten, werden generische Anwendungsprogramme genannt.

Auch die Auswertung von Objekteigenschaften kann bis zu einem gewissen Grad schemaunabhängig implementiert werden. So lassen sich beispielsweise Objektmengen entsprechend einem Auswahlkriterium bilden. Auf diese Objektmengen können dann die Funktionen einer Tabellenkalkulation angewandt werden, wobei die Objektattribute die Spalten der Tabelle definieren und für jedes Objekt eine Tabellenzeile angelegt wird. Eine andere Möglichkeit ist die Abbildung von alphanumerischen Attributinhalten auf Eigenschaften der grafischen Darstellung, wie Farbe oder Muster. Das Ergebnis solcher Auswertungen sind thematische Karten.

Bild 4 zeigt das Programm *Grobjekt*. Es basiert auf dem Laufzeitsystem und bietet somit die Möglichkeit einen logischen Gesamtdatenbestand zu bearbeiten. Durch *Grobjekt* können Objekte grafisch repräsentiert werden und Objekteigenschaften auf die Grafikattribute abgebildet werden. Als passives Werkzeug liefert es Informationen für Planungsentscheidungen im Bereich des Facility Management.

Aufgabenstellungen wie Fluchtweggestaltung, Umzugsplanung, usw. lassen sich mittels Graphen und den darauf definierbaren Algorithmen unterstützen. Daher liegt es nahe, die Graphentheorie für generische Anwendungsprogramme verfügbar zu machen. Hierzu wurde *Grobjekt* exemplarisch um eine Komponente erweitert, die in der Lage ist, die kürzesten Wege zwischen Objekten zu bestimmen. Als Ergebnis wird die Knotenlast auf die farbliche Darstellung der Objekte abgebildet, so das leicht erreichbare Objekte hell und schwer erreichbare Objekte dunkel gefärbt werden. Hierzu muß aus dem Objektmodell ein Graph extrahiert werden, der als Grundlage für die Wegeberechnung dient. Da das zugrundeliegende Objektmodell von vorn herein nicht bekannt ist, liegt beim Anwender die Verantwortung, den Graphen aus seinem Objektmodell abzuleiten. Dies ist prinzipiell möglich, da jedes Modell mit seinen Objekten und Relationen zwischen ihnen im mathematischen Sinne einen Graphen darstellt. Die Schwierigkeit besteht jedoch darin, an der Programmoberfläche in anwendergerechter Form die für die Graph-Generierung zu verwendenden Objekttypen und Relationen anzugeben.

Bewertung - Möglichkeiten und Grenzen

Generische Anwendungsprogramme sind eine Kompromißlösung bei der Bereitstellung von allgemeingültiger Funktionalität für die spezifischen Anforderungen im Bereich des Facility Management. Die Grundoperationen einer Datenbank, einer Tabellenkalkulation sowie die Funktionalität eines grafischen Editors sind wiederkehrende Anforderungen. Durch die Definierbarkeit eigener Datenstrukturen ist eine Anpassung der Software ohne programmierenden Eingriff möglich. Problematisch ist jedoch die Bereitstellung von spezifischer Funktionalität auf frei definierbaren Datenstrukturen.

Das Integrationskonzept beruht auf der Implementierung eines Laufzeitsystems, welches den Zugriff auf verteilte Datenbestände unterschiedlicher Struktur ermöglicht. Durch die Integrationswerkzeuge ist ein

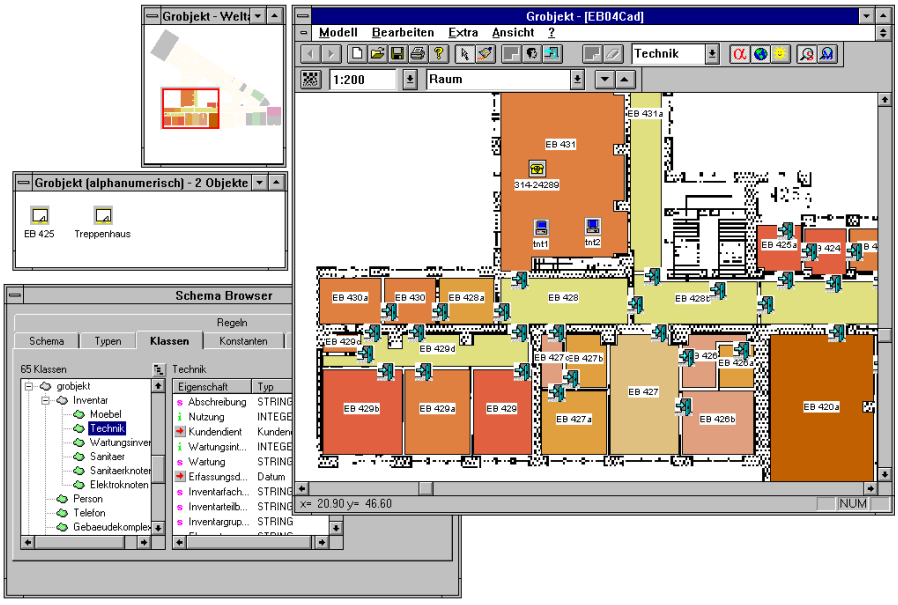

Bild 4: Anwendungsprogramm Grobjekt zur Bearbeitung eines logischen Gesamtdatenbestandes

interaktives Erstellen und Testen des logischen Gesamtdatenbestandes möglich.

Die Integration von Datenbeständen ist auf zwei Arten möglich. Bei der ersten Variante verwenden Programme das STEP-21-Dateiformat bzw. konvertieren ihre Daten in dieses Format. Diese Datenbestände können so als Quellmodelle im Modellverbund dienen. Der gewünschte Effekt des logischen Gesamtdatenbestandes wird mit der zweiten Variante erreicht. Hierbei müssen die Programme auf dem Laufzeitsystem aufsetzen.

Erfahrungen zeigen, daß durch *Schema Mapping* bei ähnlichen Strukturen schnell der Großteil der Informationen zusammengeführt werden kann. Weichen dagegen die Strukturen stark voneinander ab, wird die Zusammenführung schwierig bis unmöglich. Inwieweit Strukturunterschiede überwunden werden können, hängt von der Ausdrucksfähigkeit der Mapping-Funktionen ab. Das System wurde bewußt nicht zu einer *Mapping Language* ausgebaut, da hiermit der interaktive Charakter der Integration verletzt werden würde.

Die Identifizierbarkeit der Objekte in den Ausgangsdatenbeständen und die Überführbarkeit der Identifizierungsregeln ist ein essentielles Kriterium für das Gelingen eines Objektverbundes. Wurden bei Datenbeständen gleiche oder ähnliche Identifizierungsmechanismen verwendet, ist eine Überführung möglich - ansonsten nicht.

Die Konsistenzhaltung der Ausgangsdatenbestände ist ebenfalls essentiell. Die Struktur von Datenbeständen sowie die Regeln zur Konsistenzhaltung können durch die Modellbeschreibungssprache EXPRESS definiert werden. Häufig setzen Anwendungsprogramme jedoch weitere implizite Konsistenzregeln voraus. Verstößt ein Schreibzugriff auf das Quellmodell gegen solche Regeln, ist die Lesbarkeit der Datenbestände durch die ursprünglichen Anwendungen nicht mehr unbedingt gesichert. Der Ausweg hierfür besteht in der Bereitstellung von Methoden für die Erzeugung, Veränderung und Eliminierung von Objekten. Die standardisierte Beschreibung solcher Methoden ist jedoch noch in Entwicklung. Ob dadurch jedoch die zentralen Probleme des Produktdatenaustauschs - also die Zuordnung semantisch äquivalenter Eigenschaften und die Identifizierung gleicher Objekte der beteiligten Teilmodelle - gelöst werden können, steht zur Diskussion.

Förderung

Das Laufzeitsystem, die Integrationswerkzeuge und das Anwendungsprogramm entstanden im Projekt „Verteiltes Facility Management in Telekommunikationsnetzen“ (Kurztitel FATIMA) in Kooperation mit der HOCHTIEF Software GmbH. FATIMA ist ein Projekt im Rahmen des F&E-Programms der DeTeBerkom, einem Tochterunternehmen der Deutschen Telekom AG.

Literatur

ISO 10303 Product Data Representation and Exchange - Part 11: The EXPRESS-Language Reference Manual, Part 21: Clear Text Encoding of the Exchange Structure, Part 22: Standard Data Access Interface

Kim, Won: Modern Database Systems - The Object Model, Interoperability, and Beyond. Addison-Wesley, 1995

Kolbe, Ranglack, Steinmann: „Ein Laufzeitsystem für dynamische Objektstrukturen für Entwurfsanwendungen“ in diesem Band

Pfennigschmidt, Kolbe, Pahl: „Integration von Datenmodellen - Eine Methodik zum Produktdatenaustausch“ in diesem Band