SIGMA$_{Demo}$ - Eine FDBS-Demonstrationssoftware*

Guido Grohmann Michael Höding

Otto-von-Guericke-Universität Magdeburg
Institut für Technische Informationssysteme
Postfach 4120, D–39016 Magdeburg
E-mail: {hoeding}@iti.cs.uni-magdeburg.de
Tel.: ++49-391-67-12220 Fax: ++49-391-67-12020

Kurzfassung

In diesem Beitrag geben wir einen Überblick zu der für ein konkretes Beispielszenario realisierten FDBS-Demonstrationssoftware SIGMA$_{Demo}$. Ziel dieser Softwareentwicklung ist es, Probleme und Möglichkeiten der Konsistenzsicherung in föderierten Systemen an einem praktischen Beispiel zu verdeutlichen. Dazu beschreiben wir sowohl die lokalen Schemata als auch das integrierte Schema des FDBS. Weiterhin gehen wir auf Implementierungsaspekte sowie auf entwickelte lokale und globale Applikationen ein.

1 Einleitung

Im Rahmen des vom Land Sachsen-Anhalt geförderten Projektes SIGMA$_{FDB}$ [HTJ$^+$95] wird eine Demonstrationssoftware zur Verdeutlichung der Probleme und Lösungsansätze *föderierter Datenbanksysteme (FDBS)* ([SL90], [HM85]) entwickelt. Diese Software mit der Bezeichnung SIGMA$_{DEMO}$ stellt ein Beispiel eines FDBSs anhand eines produzierenden Unternehmens dar. Die SIGMA$_{DEMO}$-Software entstand mit dem Ziel, ein praktisches Anschauungsobjekt zur Verdeutlichung der Probleme bei Entwurf und Betrieb föderierter Systeme zu erhalten. Durch eine nunmehr mögliche experimentelle Nutzung der Demonstrationssoftware können Probleme und Potential der Integritätssicherung in föderierten Systemen auch von einem eher praxisorientierten Standpunkt untersucht und verdeutlicht werden.

In diesem Beitrag gehen wir zunächst auf das Beispielszenario ein und stellen die zu integrierenden lokalen Schemata vor. Der darauffolgende Abschnitt stellt das Ergebnis einer durchgeführten Schemaintegration dar. Danach werden Implementierungsaspekte diskutiert. Nach der Vorstellung zweier Applikationen, die auf der globalen Schnittstelle des FDBS arbeiten, geben wir einen kurzen Ausblick auf zukünftige Arbeiten.

2 Das Beispielszenario

In diesem Abschnitt beschreiben wir die im Rahmen der SIGMA$_{Demo}$ integrierten lokalen Informationssysteme und das zugrunde liegende Beispielszenario. Die Grundlage der Betrachtungen bildet ein (fiktives) produzierendes Unternehmen, das komplette Materialflußanlagen für Stückgüter anbietet. Zum Transport der Stückgüter werden Rollenförderer verwendet. Das Tätigkeitsfeld des Unternehmens umfaßt die Produktion von Rollenförderern und den Vertrieb kompletter Materialflußanlagen, deren Bestandteile eigenproduzierte und auch fremdgefertigte Transportmaschinen, wie z.B. Gabelstapler, sind.

Innerhalb des Unternehmens gibt es eine Konstruktions- und eine Vertriebsabteilung, die jeweils über eigene Datenhaltungssysteme verfügen. Von der Konstruktionsabteilung werden Rollenförderer (`Roller_Conveyor, RC`) konstruiert, sowie der zugehörige Ablauf der Produktion vorbereitet. Wenn ein

*Diese Arbeit wurde gefördert vom Land Sachsen-Anhalt unter FKZ: 1987A/0025 "Föderierung heterogener Datenbanksysteme und lokaler Datenhaltungskomponenten zur systemübergreifenden Integritätssicherung" – "**SIGMA**$_{FDB}$ (**S***chema* **I***ntegration and* **G***lobal integrity* **M***aintenance* **A***pproach for* **F***ederated* **D***ata* **B***ases*)"

solcher Rollenförderer einen eigenen Antrieb besitzt, müssen weitere technische Daten in der Konstruktionsdatenbank berücksichtigt werden. Daher werden angetriebene (`Driven_Roller_Conveyor, DRC`) und nichtangetriebene Rollenförderer unterschieden.

Die beim Entwurf oder bei der Konstruktion gewonnenen und benötigten Daten werden von einer CAD/CAP-Anwendung in einer CAx-Datenbank verwaltet. In der Konstruktionsabteilung ist hierzu ein objektorientiertes Datenbanksystem zur Verwaltung von CAD-Daten und anderen technischen Daten von Rollenförderern als Komponenten kompletter Materialflußanlagen etabliert. Eine hier vorhandene Datenbankanwendung erlaubt Erfassung und Manipulationen der gespeicherten Daten.

Die Vertriebsabteilung benutzt ein relationales Datenbanksystem. In dieser Datenbank sind u.a. alle aktuell produzierten Rollenförderer und alle anderen von der Firma angebotenen Transportmaschinen mit ihren Preisen und Lieferfristen erfaßt. Ein Anwendungsprogramm erlaubt auch hier Einfügen, Löschen und Ändern von Datensätzen in der Verkaufsdatenbank.

Aus den bis jetzt beschriebenen Sachverhalten bezüglich Tätigkeitsfeld und Unternehmensstruktur ist ersichtlich, daß insbesondere zu bereits fertig konstruierten Rollenförderern Daten zu in der Realwelt identischen Objekten in den verschiedenen Datenbanken gespeichert sind. Die Datenbankschemata (genauer: Sichten auf die Schemata) dieser Systeme sollen nun etwas näher betrachtet werden.

Das lokale Schema der CAx-Datenbank

Das objektorientierte Datenbankschema der CAx-Datenbank ist in Abbildung 1 grafisch dargestellt. Aus dieser Abbildung sind die Intensionen der Klassen des Schemas der OODB zu entnehmen. Außerdem wird die (einfache) Vererbungshierarchie verdeutlicht.

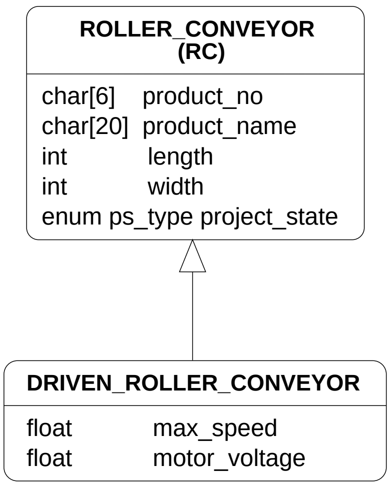


Abbildung 1: CAx-Schema

Alle Integritätsbedingungen werden innerhalb der für die Eintragung von Attributwerten in ein Objekt zuständigen Klassenmethoden überprüft. Im Fehlerfall werden die Attributwerte nicht eingetragen. Die Daten werden im Objekt gekapselt.

Das Schema der relationalen Verkaufsdatenbank

In der relationalen Datenbank der Vertriebsabteilung wird eine zu veräußernde Transportmaschine mit ihrer Produktnummer, einer Bezeichnung, dem Preis, der Lieferdauer und einem weiteren Attribut, das eine Klassifikation der gespeicherten Transportmaschinen ermöglicht, abgelegt. Das Schema der relationalen Verkaufsdatenbank wurde in einer für relationale Datenbanksysteme gebräuchlichen DDL notiert und ist folgendermaßen aufgebaut:

```
Create Table transport_machine (
  product_no char(6) primary Key,
```

```
product_name vchar(40),
price number(10,2),
delivery_period number(3),
machine_class vchar(30)) );
```

Die Bezeichnung (oder der Name) einer Transportmaschine wird in der Verkaufsdatenbank stets in großen Buchstaben notiert, während in der CAx-Datenbank auch kleine Buchstaben bei der Benennung zugelassen sind. Auch in diesem Schema ist die Produktnummer ein Schlüssel. Im Gegensatz zum objektorientierten Datenbanksystem der Konstruktionsabteilung werden zur Überwachung der Schlüsselbedingung Konsistenzsicherungsmechanismen des Datenbanksystems benutzt, wie das Schlüsselwort `primary key` erkennen läßt. Das Attribut Preis ist ein Gleitkommawert, während die Lieferzeit in vollen Tagen anzugeben und somit ganzzahlig ist. Das Attribut `Maschinenklasse` muß bei einer eigenproduzierten Transportmaschine einen vordefinierten Wert haben. Ein Beispiel für eine solche vordefinierte Maschinenklasse ist `RC`, was einem nichtangetriebenen Rollenförderer entspricht. Die bis jetzt beschriebenen Datenbankschemata und Applikationen sind nun Ausgangspunkt für eine vom Unternehmen gewünschte Integration, deren Ergebnis, ein integriertes globales Schema, im folgenden Abschnitt beschrieben wird.

Das integrierte globale Schema

Das im SIGMA$_{DEMO}$-Projekt realisierte Beispiel wurde zu wesentlichen Teilen [CHJ+96] entnommen. Die zur Gewinnung eines integrierten Schemas durchgeführten Schematransformationen und die Schemaintegration werden hier nicht näher geschrieben. Es wird nur amgemerkt, daß das in [Sch95, SS96] für den FDBS-Entwurf vorgeschlagene semantisch arme Datenmodel GIM (*Generic Integration Model*) zur Erstellung des integrierten Schemas für den SIGMA$_{Demo}$-Prototyp Verwendung fand.

Als Ergebnis der auf Basis von GIM durchgeführten Schematransformation und -integration wurde das folgende objektorientierte Schema abgeleitet, welches Mehrfachvererbung verwendet.

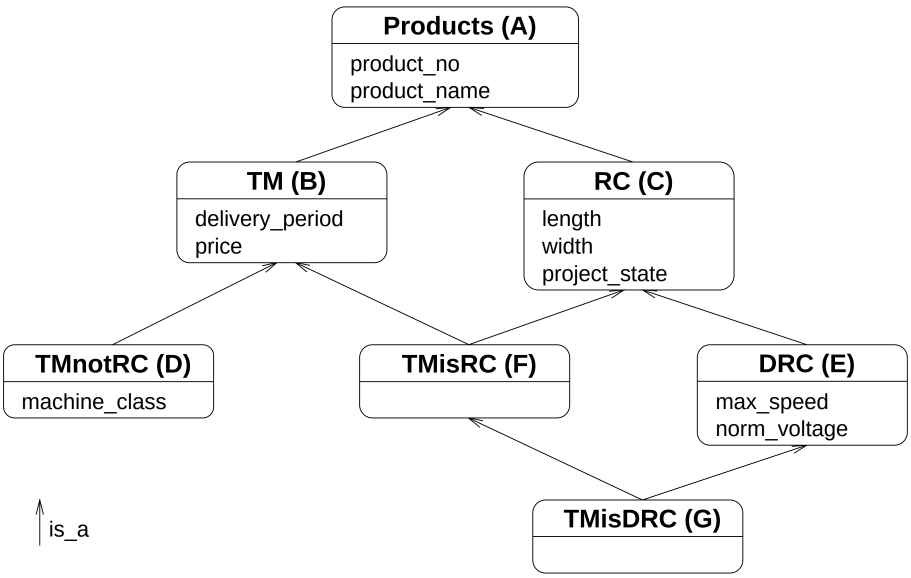


Abbildung 2: Abgeleitetes objektorientiertes Schema

Die Extensionen und Intensionen der in Abbildung 2 dargestellten Klassen werden nun im Einzelnen erläutert.

AC: (Produkt) - eine abstrakte Klasse. Die Intension dieser Klasse enthält die bei jedem Produkt vorhandene Produktnummer und eine Produktbezeichnung. Die Produktnummer ist ein Schlüssel, d.h. es existiert hierfür eine globale Schlüsselbedigung. Die Bezeichnung eines Produktes ist auf eine Länge von 20 Zeichen begrenzt worden, wie es bereits im objektorientierten Datenbankschema der CAx-Datenbank der Fall war. Bei Manipulationen an Produktbezeichnungen bestehender Datensätze sind aufgrund der in der Verkaufsdatenbank (in der Relation `Transport_Maschine`) geforderten Großschreibung Zeichenkettenkonvertierungen notwendig.

RC: Die Extension dieser Klasse enthält alle Konstruktionsdaten von Rollenförderern, zu denen keine Einträge in der Verkaufsdatenbank existieren. In der Intension dieser Klasse finden sich neben den ererbten Attributen und Integritätsbedingungen die geometrische Daten enthaltenden Attribute Länge und Breite wieder. Die bereits im lokalen Schema der Konstruktionsdatenbank definierten Integritätbedingung für diese beiden Attribute gibt es hier auch. Der Projektstatus eines Rollenförderers mit seiner transitionalen Integritätsbedingung komplettiert die Intension dieser Klasse.

DRC: Die Extension dieser Klasse besteht aus allen angetriebenen Rollenförderern, zu denen nur Konstruktionsdaten verwaltet werden. Die Intension unterscheidet sich von der Intension der Klasse `RC` durch die neu hinzugekommenen Attribute max_speed und motor_voltage. Diese Attribute stammen von der Klasse Driven_Roller_Conveyor aus dem lokalen Schema der CAx-Datenbank. Sie beschreiben die Höchstgeschwindigkeit und die Höhe der Versorgungsspannung des Antriebs. An diese Attribute sind keine Integritätsbedingungen geknüpft.

TM: Hier finden sich die Transportmaschinen aus der Verkaufsdatenbank wieder, aus deren Maschinenklasse ersichtlich ist, daß sie Rollenförderer sind. Sie werden mit ihrem Preis, der Lieferzeit und der Maschinenklasse verwaltet. Die Produktnummer und die Bezeichnung erben sie - wie alle anderen Klassen auch - von der abstrakten Klasse AC.

TMnotRC: Die Intension dieser Klasse gleicht der der vorangegangenen Klasse, mit Ausnahme der Bedingung, daß Transportmaschinen, die aufgrund ihrer Maschinenklasse *nicht* in der Konstruktionsdatenbank vertreten sein können, in diese Klasse gehören. Das Attribut Maschinenklasse darf also *nicht* den Wert RC oder DRC enthalten. Die Extension enthält also alle Transportmaschinen, die *nur* in der Verkaufsdatenbank verwaltet werden können.

TM_RC: Die Intension dieser Klasse enthält zunächst alle Attribute und Bedingungen der Klasse TM. Alle Attribute der Klasse RC, mit Ausnahme von Produktnummer und Bezeichnung, die ja schon vorhanden sind, kommen dazu. Das gilt natürlich auch für die Integritätsbedingungen, wie die Schlüsselbedingung der Produktnummer und die Längen-Breiten-Bedingung oder die transitionale Bedingung für den Projektstatus. Folgerichtig befinden sich in der Extension dieser Klasse all die Rollenförderer ohne Antrieb, zu welchen Konstruktionsdaten *und* Verkaufsdaten verfügbar sind. Implizit ergibt sich, daß die Maschinenklasse eines solchen Objektes immer den Wert `RC` hat.

TMisDRC: Diese Klasse ist der vorangegangenen Klasse sowohl bezüglich Extension als auch Intension recht ähnlich. Der Unterschied ist nur, daß hier angetriebene Rollenförderer verwaltet werden. Die Intension enthält daher zusätzlich zu den Attributen und Bedingungen der Klasse TM_RC die von der Klasse DRC ererbten Attribute maxspeed und motor_voltage. Es ergibt sich aus den vorrangegangenen Fakten die Restriktion, daß als Maschinenklasse `DRC` eingetragen sein muß.

Basierend auf diesem objektorientierten Schema wurde nun die Prototypentwicklung vorgenommen. Die oben beschriebenen Schemaklassen sind in der Föderierungsschicht der Demonstrationssoftware wiederzufinden. Ihre Funktion wird bei der Benutzung des im nächsten Abschnitt beschriebenen Admintools demonstriert.

3 Implementation der SIGMA$_{DEMO}$-Software

Dieser Abschnitt gibt einen kurzen Überblick über die Softwarekomponenten von SIGMA$_{DEMO}$. Alle hier beschriebenen Programme wurden in C++ auf Sun-Workstations geschrieben, mit Ausnahme der Komponenten, die embedded SQL verwenden und in C programmiert wurden. Zunächst wird eine Übersicht über die einzelnen Bestandteile gegeben:

- lokale Informationssysteme der Abteilungen:

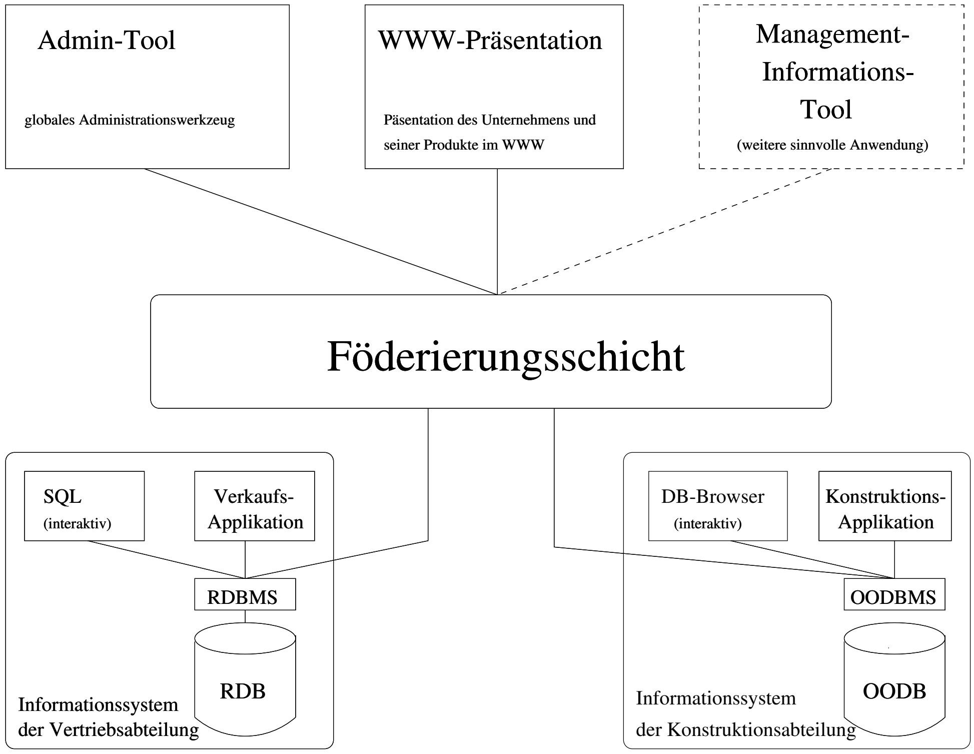


Abbildung 3: Übersicht über die Softwarekomponenten von SIGMA$_{DEMO}$

- Informationssystem der Konstruktionsabteilung: Verwendung eines objektorientierten Datenbanksystems zur Speicherung der Produkt- und Projektdaten mit einer eigenen lokalen Datenbankanwendung und ggf. eines vorhandenen Datenbankbrowsers.
- Informationssystem der Verkaufsabteilung: Verwaltung der Verkaufsdaten unter Verwendung eines RDBSs mittels einer eigens dafür geschriebenen Applikation sowie der interaktiven SQL-Schnittstelle.

- die Föderierungsschicht: die Klassen des integrierten Schemas, auf deren Basis einfach und schnell globale Anwendungen erstellt werden können.
- Anwendungen mit globaler Sicht auf die Unternehmensdaten:
  - ein Administrationstool, mit dem Objekte der integrierten Schemaklassen erzeugt, angezeigt und manipuliert werden können,
  - eine Präsentation des Unternehmens im WWW inklusive einer Datenbankanfrage mit globaler Sicht auf die Daten beider Abteilungen.

In den nun folgenden Abschnitten werden die wichtigsten dieser Programme mit ihrem Funktionsumfang und Leistungsvermögen vorgestellt.

3.1 Die Föderierungsschicht

In der Föderierungsschicht wurden die Klassen des im vorangegangenen Abschnitt vorgestellten integrierten Schemas implementiert. Die Implementierung wurde so einfach wie möglich gehalten. Um die auf der Basis der Föderierungsschicht erstellten Anwendungen, die eine globale Sicht auf alle föderierten Daten haben, immer über die aktuellen lokalen Datenbestände zu informieren, werden Objektlisten der betreffenden Klasse des integrierten Schemas für die Dauer einer globalen Operation temporär im Haupsteicher gehalten. Diese Pufferung wurde in sog. Metaklassen implementiert. Jede Schemaklasse hat demzufolge eine Metaklasse. Hier sind die Operationen zur Verwaltung der eingangs erwähnten Listen untergebracht. Ein Konstruktor baut eine Liste der Objekte der gewünschten Klasse auf. Zum Einfügen, Ändern und Löschen globaler Objekte sind jeweils eigene Methoden implementiert, die:

- Operationen auf ggf. beiden beteiligten Datenbanken vornehmen,
- die Verletzung von Integritätsbedingungen anzeigen sowie
- die Reaktion einer globalen Anwendung auf eine zu erwartende Verletzung der globalen Schlüsselbedingung unterstützen.

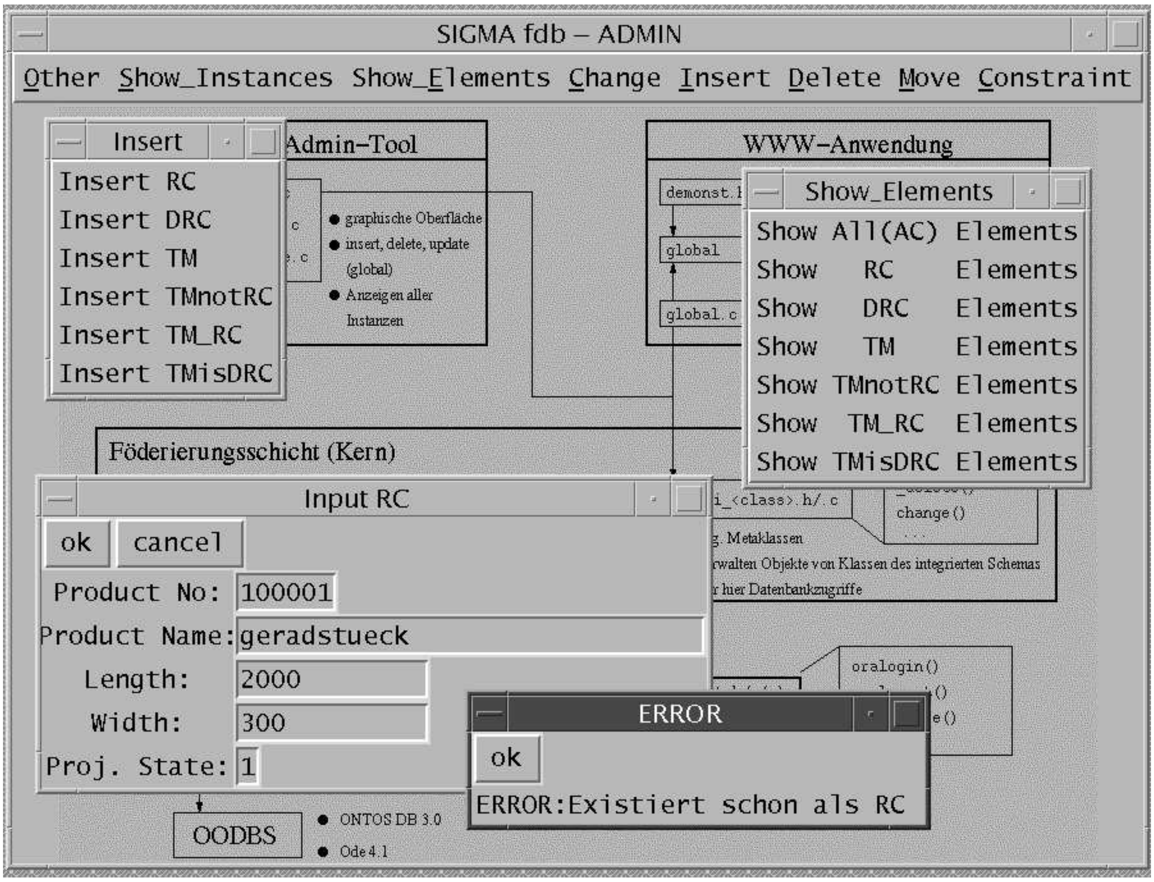


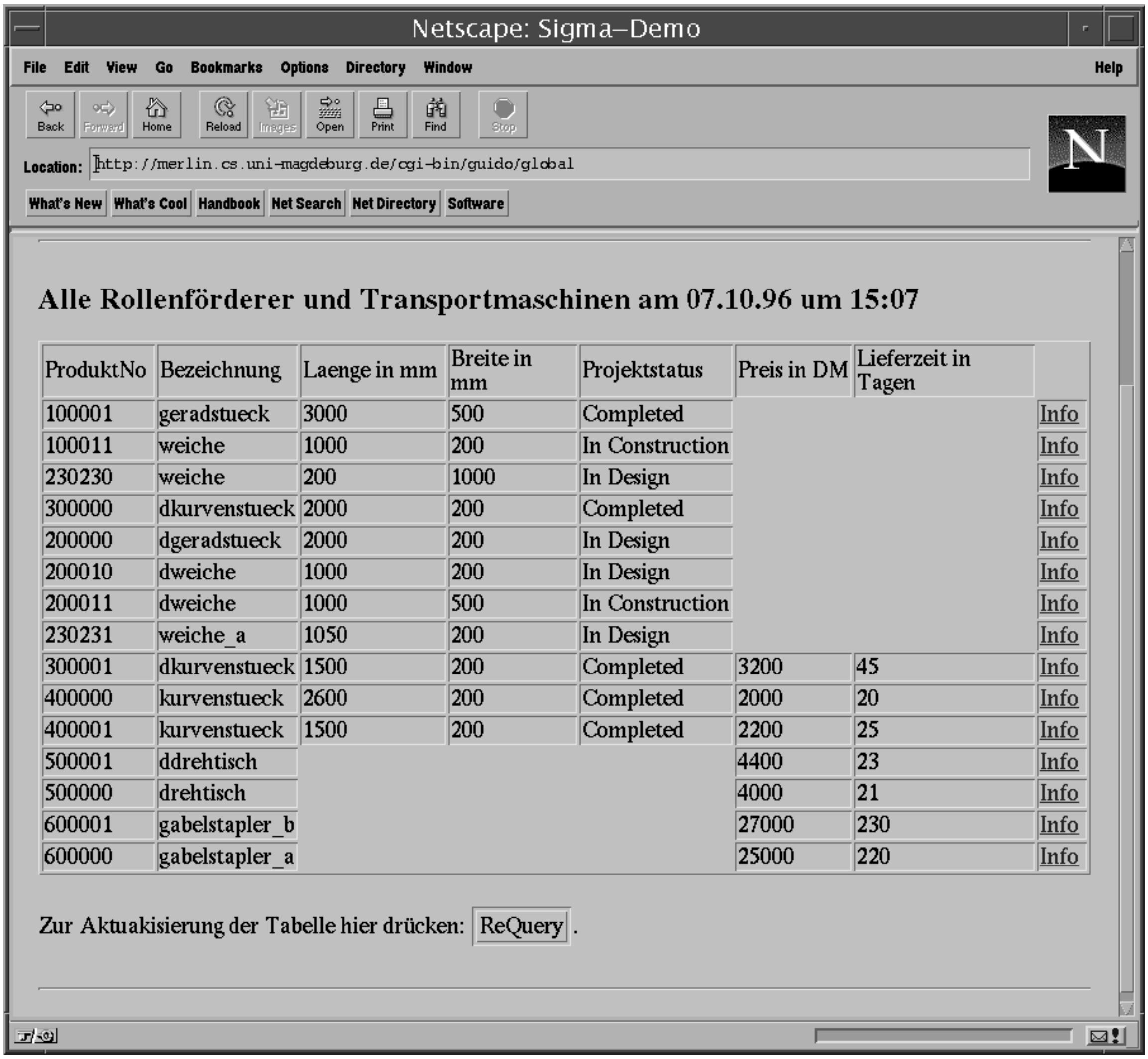
Abbildung 4: Das Admintool im Einsatz

Dadurch können globale Anwendungen, die globale Integritätsbedingungen unterstützen, relativ effizient erzeugt werden. Mit der Programmierung des Admintools wurde dies anschaulich demonstriert. Werden die durch die Föderierungsschicht angebotenen Mittel zur Verhinderung globaler Inkonsistenzen nicht benutzt, ist z.B. eine Verletzung der globalen Schlüsselbedingung möglich. Lokal zu erkennende Verstöße gegen Integritätsbedingungen werden durch die lokalen Datenbanksysteme oder Klassenmethoden unterbunden.

3.2 Das Admin-Tool

Mit dem SIGMA$_{DEMO}$-Admintool können globale Datenbankoperationen vollzogen werden. Es nutzt alle im vorangegangenen Abschnitt dargelegten Möglichkeiten aus. Das Admintool wurde mit einer grafischen Benutzeroberfläche ausgestattet.

Mittels dieses Tools können zur Darstellung der momentanen Datenbestände die Extensionen der (globalen) Schemaklassen durchiteriert werden. Neueintragungen und Änderungen von Datensätzen unter Einhaltung globaler Bedingungen sind hier ebenfalls realisiert. Darüber hinaus wird unter Verwendung der in den Metaklassen bereitgestellten Methoden ein globales Verschieben entlang der Vererbungsstruktur des globalen Schemas angeboten, so daß gezielte Änderungen der Klassenzugehörigkeit einzelner Objekte und die daraus resultierenden lokalen Datenbankoperationen unter Verwendung der beiden lokal implementierten Anwendungsprogramme sichtbar gemacht werden können.



Alle Rollenförderer und Transportmaschinen am 07.10.96 um 15:07

ProduktNo	Bezeichnung	Laenge in mm	Breite in mm	Projektstatus	Preis in DM	Lieferzeit in Tagen	
100001	geradstueck	3000	500	Completed			Info
100011	weiche	1000	200	In Construction			Info
230230	weiche	200	1000	In Design			Info
300000	dkurvenstueck	2000	200	Completed			Info
200000	dgeradstueck	2000	200	In Design			Info
200010	dweiche	1000	200	In Design			Info
200011	dweiche	1000	500	In Construction			Info
230231	weiche_a	1050	200	In Design			Info
300001	dkurvenstueck	1500	200	Completed	3200	45	Info
400000	kurvenstueck	2600	200	Completed	2000	20	Info
400001	kurvenstueck	1500	200	Completed	2200	25	Info
500001	ddrehtisch				4400	23	Info
500000	drehtisch				4000	21	Info
600001	gabelstapler_b				27000	230	Info
600000	gabelstapler_a				25000	220	Info

Zur Aktuakisierung der Tabelle hier drücken: ReQuery .

Abbildung 5: Globale Datenbankanfrage mit einem WWW-Browser

3.3 Ein WWW-Produktkatalog

Die Präsentation eines Unternehmens mittels eines WWW-Browsers soll einen Überblick der angebotenen Transportmaschinen liefern. Um dieses Angebot möglichst immer aktuell anzubieten, wurde unter Zuhilfenahme der Föderierungschicht eine globale Anfrage realisiert.

Durch eine WWW-Anfrage kann jederzeit eine Übersicht erzeugt werden, die alle z.Z produzierten oder verkauften Transportmaschinen zeigt. Dazu wird mittels eines CGI-Skriptes eine globale Anfrage, d.h. das Durchiterieren aller integrierten Schemaklassen, durchgeführt. Aus der dann generierten Tabelle können durch Auswahl der betreffenden Links weitere Daten (Datenblätter) abgerufen werden. Die Zusammenstellung dieser Datenblätter erledigt ein weiteres CGI-Skript.

4 Zusammenfassung und Ausblick

In dem vorliegenden Beitrag wurde eine Demonstrationssoftware vorgestellt, die ein FDBS auf der Grundlage eines praxisnahen Beispiels realisiert. Ausgehend von einem fiktiven produzierenden Unternehmen mit als gegeben vorrausgesetzten heterogenen Datenbanksystemen wurde das Ergebnis einer beispielhaft durchgeführten Schemaintegration präsentiert und implementiert. Die Implementation wurde bezüglich der verwendeten Datenbanksysteme flexibel gestaltet und befindet sich in einem anwendungsbereiten Zustand.

In der aktuellen Version kann die Software SIGMA$_{DEMO}$ für Vorführungen und Experimente genutzt werden. Bei der Implementation wurde darauf geachtet, daß das Ergebnis offen für geplante Erweiterungen ist. Eine vorgesehene Erweiterung ist die Realisierung von Integritätssicherungsmaßnahmen unter Verwendung aktiver Mechanismen[Tür96].

Literatur

[CHJ+96] S. Conrad, M. Höding, S. Janssen, G. Saake, I. Schmitt und C. Türker. Integrity Constraints in Federated Database Design. Preprint Nr. 2, Fakultät für Informatik, Universität Magdeburg, April 1996.

[HM85] D. Heimbigner und D. McLeod. A Federated Architecture for Information Management. *ACM Transactions on Office Information Systems*, 3(3):253–278, Juli 1985.

[HTJ+95] M. Höding, C. Türker, S. Janssen, K.-U. Sattler, S. Conrad, G. Saake und I. Schmitt. Föderierte Datenbanksysteme — Grundlagen und Ziele des Projektes **SIGMA**$_{FDB}$. Preprint Nr. 12, Fakultät für Informatik, Universität Magdeburg, Dezember 1995.

[Sch95] I. Schmitt. Flexible Integration and Derivation of Heterogeneous Schemata in Federated Database Systems. Preprint Nr. 10, Fakultät für Informatik, Universität Magdeburg, November 1995.

[SL90] A. P. Sheth und J. A. Larson. Federated Database Systems for Managing Distributed, Heterogeneous, and Autonomous Databases. *ACM Computing Surveys*, 22(3):183–236, September 1990.

[SS96] I. Schmitt und G. Saake. Schema Integration and View Generation by Resolving Intensional and Extensional Overlappings. In K. Yetongnon und S. Hariri, Herausgeber, *Proc. of the 9th ISCA Int. Conf. on Parallel and Distributed Computing Systems (PDCS'96), Dijon, France*, S. 751–758, September 1996.

[Tür96] C. Türker. An Approach to Supporting Semantic Integrity of Federated Databases. In *Proc. of the Int. Workshop on Heterogeneous Information Management, Prague, Czech Republic, November 1996*, 1996. *To appear.*