Föderierung heterogener Datenbanksysteme und lokaler Datenhaltungskomponenten zur systemübergreifenden Integritätssicherung - Kurzvorstellung des Projektes SIGMA$_{FDB}$ - (Kurzbeitrag)

Gunter Saake Andreas Christiansen Stefan Conrad
Michael Höding Ingo Schmitt Can Türker

Otto-von-Guericke-Universität Magdeburg
Institut für Technische Informationssysteme
Postfach 4120, D–39016 Magdeburg

Zusammenfassung. In diesem Beitrag geben wir einen Überblick über das Projekt **SIGMA**$_{FDB}$, welches sich mit der Entwicklung einer Basis-Informationsinfrastruktur als Grundlage einer integrierten und einheitlichen Datenhaltung für alle Phasen der Fabrikplanung befaßt. Dazu soll ein föderiertes heterogenes Informationssystem entstehen, das als Rahmensystem zur Integration aller an der Fabrikplanung beteiligten Software-Werkzeuge einschließlich deren lokaler Datenbestände dient. Damit sollen bislang separate Werkzeuge, die für Produktentwurf, Produktionsplanung und Fabriksimulation eingesetzt werden oder dafür noch entwickelt werden, synergetisch zusammengefügt werden.
Zentrale Aufgabe eines solchen föderierten Informationssystems ist neben der Bereitstellung einer homogenen Datenbankschnittstelle für globale Anwendungen die systemübergreifende Gewährleistung der Datenkonsistenz. Die Verbindung der Einzelsysteme verlangt die Integration der verschiedenen Datenschemata, welche oft auf unterschiedlichen Datenmodellen basieren.

1 Einleitung

Datenbankbasierte Anwendungen repräsentieren einen festen Bestandteil moderner informationsverarbeitender Systeme. Insbesondere die Verfügbarkeit relationaler Datenbanksysteme erschloß neue Anwendungsfelder jenseits des klassischen Datenbankeinsatzes in Banken, Versicherungen oder Bibliotheken. Heute erstrecken sich die Anwendungsgebiete von datenbankbasierten Anwendungen bis in klein- und mittelständige Unternehmen. Neben Datenbanksystemen entstanden zahlreiche Systeme, welche spezielle Anwendungsfelder abdecken. Beispiele hierfür sind CAD–Systeme, Produktplanungssysteme oder auch Systeme zur rechnergestützten Produktion. Zur Speicherung der bereichsspezifischen Daten verwenden diese Systeme Datenbanken oder komplex strukturierte Datenhaltungssysteme, welche auf Dateisystemen aufsetzen. Moderne Unternehmensstrukturen fordern insbesondere eine Überlappung der produktionsvorbereitenden Unternehmensbereiche. Diese scheinbar effizienzsteigernde Überlappung von

* **SIGMA**$_{FDB}$ (**S**chema **I**ntegration and **G**lobal integrity **M**aintenance **A**pproach for **F**ederated **D**ata **B**ases), gefördert durch das Land Sachsen-Anhalt unter FKZ 1987 A / 0025

Unternehmensbereichen bedingt aus informationstechnischer Sicht eine redundante und inkonsistente Datenhaltung, welche einen erhöhten Kostenbedarf induziert.

Die Verwaltung der Daten auf Basis einer zentralen integrierten Datenbank erscheint aufgrund der unbeherrschbaren Komplexität und der Aufgabe, die lokale *Autonomie* der Teilsysteme zu erhalten, unrealistisch. Heterogenitätsformen auf Hardware-, Datenmodell-, Datenbankmanagementsystem-, Schema- und Datenebene sind eine direkte Folge der Autonomie. Zur Beherrschung der verschiedenen Formen der Heterogenität ist es sinnvoll, ein föderiertes Datenbanksystem einzusetzen, welches unter Wahrung der lokalen Autonomie die Daten aller an der Föderation beteiligten Systeme verwaltet. Der wesentliche Vorteil dieses Ansatzes besteht darin, daß das fachgebietsspezifische Know-how, welches in den einzelnen Systemen angesammelt wurde, direkt unter dem Gesichtspunkt der Investitionssicherung weiterverwendet werden kann. Ferner werden Risiken, wie sie bei der Umstellung von Insellösungen auf ein monolithisches Gesamtsystem auftreten, vermieden.

Das Projekt **SIGMA**$_{FDB}$ ist Bestandteil des Forschungsschwerpunktes *Neue Produktionssysteme - Experimentelle Fabrik* des Landes Sachsen-Anhalt. Im Rahmen dieses Projektes erarbeiten Informatiker, Maschinenbauer und Geisteswissenschaftler die Grundlagen einer zukunftsorientierten Fabrikplanung, welche vor allem durch den Einsatz modernster Informationstechnologie geprägt sein wird. Der Beitrag des Projektes **SIGMA**$_{FDB}$ besteht in der Erarbeitung der Grundlagen föderierter Datenbanken. Dazu gehören die Themengebiete

- Konsistenzsicherung in föderierten Datenbanken,
- Schematransformation und -integration,
- Integration von Dateien in Datenbankföderationen,
- Sicherheit in föderierten Datenbanken und
- Transaktionsverwaltung in föderierten Datenbanken.

2 Grundlagen

In diesem Kapitel werden wichtige Begriffe und Grundlagen föderierter Datenbanken [SL90] vorgestellt.

Ein *föderiertes Datenbanksystem* (FDBS) besteht aus dem *föderierten Datenbankmanagementsystem* (FDBMS) und aus einem oder mehreren *Komponentendatenbanksystemen* (CDBS) (vgl. Abb. 1). Das FDBMS wird oft auch als Föderierungsdienst oder Föderierungsschicht bezeichnet [HM85]. Das FDBS dient dazu, logisch und möglicherweise physisch verteilte heterogene Datenbanksysteme bei weitgehender Erhaltung der lokalen Autonomie der CDBS zu föderieren.

Die CDBS können dabei hinsichtlich der Laufzeitumgebung, des Datenmodells und der Datenbankschemata heterogen sein [SL90]. Auf der lokalen Schnittstelle der CDBS arbeiten lokale Applikationen, die auf die speziellen Gegebenheiten der Laufzeitumgebung, des unterstützten Datenmodells und der realisierten lokalen Schemata zugeschnitten sind. Das FDBS operiert ebenfalls auf den

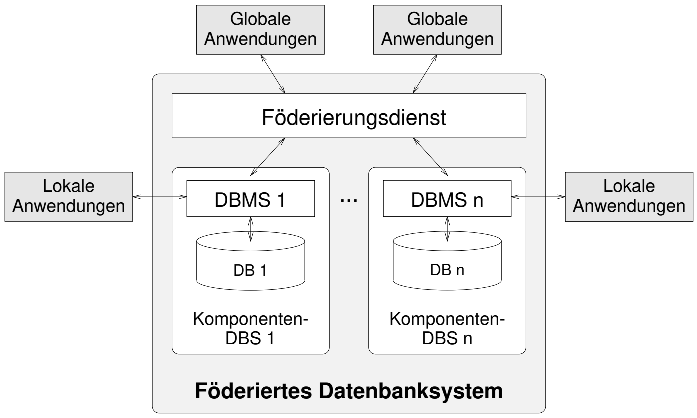

Abb. 1. Komponenten eines föderierten Datenbanksystems

lokalen Schnittstellen der einzelnen CDBS. Bei vollständiger Wahrung der Autonomie der CDBS verhält sich der Föderierungsdienst wie eine lokale Applikation.

Die wesentliche Aufgabe eines FDBMS besteht darin, eine globale homogene Schnittstelle für den einheitlichen und transparenten Zugriff auf föderierte Daten bereitzustellen. Zu den weiteren Aufgaben des Föderierungsdienstes gehören die Überwindung der Heterogenität, die systemübergreifende Transaktionsverwaltung und Konsistenzsicherung sowie die Bereitstellung einer Anfragesprache.

Es werden ein oder mehrere föderierte Schemata für den einheitlichen und transparenten Zugriff auf alle föderierten Daten realisiert. Die Darstellung des föderierten Schemas basiert auf einem einheitlichen Datenmodell, dem sogenannten *kanonischen Datenmodell* [CSGS94]. Über externe Schnittstellen können globale Applikationen die Dienste des föderierten Datenbanksystems nutzen.

In [SL90] wird eine 5-Schema-Ebenen-Referenz-Architektur vorgestellt, die sich für die vergleichende Bewertung verschiedener Architekturen föderierter Datenbanksysteme und in der FDBS-Entwicklung durchgesetzt hat. Wegen der grundlegenden Bedeutung dieser Architektur werden wir diese im folgenden kurz vorstellen. Wir setzen dabei voraus, daß für die Föderierungsschicht ein geeignetes kanonisches Datenmodell existiert.

Die konzeptionellen Schemata der an der Föderation beteiligten Komponentendatenbanken (CDB) bilden die lokalen Schemata des FDBS. Sie basieren auf dem Datenmodell des jeweiligen DBMS. Die lokalen Schemata werden in das kanonische Datenmodell überführt. Dadurch wird die Heterogenität hinsichtlich des genutzten Datenmodells überwunden. Es enstehen die Komponentenschemata. Durch eine Filterung der an der Föderation beteiligten Schemateile werden die Exportschemata erzeugt. Die Überwindung der Schemaheterogenität erfolgt durch die Integration aller Exportschemata zu einem globalen föderierten Schema. Dabei mögliche Integrationsstrategien werden u.a. in [BLN86] diskutiert. Bei der Schemaintegration treten die bereits aus der Sichtenintegration bekannten Konflikte auf, die nicht immer vollständig lösbar sind. Aus dem föderierten Schema können externe Schemata abgeleitet werden. Dabei ist nicht nur die Generierung von Schemaheterogenität, sondern auch von Datenmodellheterogenität möglich. Damit besteht die Möglichkeit, Schemata anzubieten, die äquivalent den lokalen Schemata sind, wodurch die Migration von lokalen Applikationen vom lokalen Datenbanksystem auf das föderierte Datenbanksystem durchführbar wird [RS94].

Nicht alle Schemaebenen müssen in der Realisierung eines FDBS ausgeprägt sein. So ist es beispielsweise möglich, die Filterung der zu föderierenden Schemateile bereits durch Nutzung externer Schemata des lokalen DBMS zu realisieren. Dadurch kann eine größere Entwurfsautonomie der CDBS unterstützt werden.

3 Projektziele und Lösungsansätze

Hauptziel des Projektes ist es, die informationstechnischen Grundlagen zur Schaffung einer zukunftsorientierten Unternehmens-, Informations- und Kommunikationsinfrastruktur zu erarbeiten. Dabei stehen die Unterstützung eines parallelen Arbeitens und die Umsetzung der Forderung nach kürzeren Innovationszyklen im Mittelpunkt. Das parallele Arbeiten bedingt eine globale Konsistenzsicherung, wobei jedoch Restriktionen hinsichtlich der lokalen Autonomie so weit wie möglich vermieden werden müssen. Da die im vorangegangenen Kapitel vorgestellte Architektur diese Anforderungen erfüllt, bietet der Einsatz von FDBS ein geeignetes und sinnvolles Mittel zur Lösung obiger Probleme. Aufgrund der Komplexität der Aufgabenstellung beschränken wir uns im Rahmen des Projektes **SIGMA**$_{FDB}$ auf Szenarien aus dem Bereich der Fabrikplanung.

3.1 Konsistenzsicherung durch aktive Mechanismen

Ein grundlegender Aspekt beim Zusammenschluß mehrerer heterogener Systeme ist die Sicherung der Konsistenz der föderierten Daten. Während die Konsistenzsicherung in den einzelnen CDBS von dem jeweiligen lokalen DBMS realisiert wird, existieren zunächst keine Mechanismen, die eine globale Konsistenzsicherung erlauben. Die bei der Umsetzung eines integritätssichernden Konzeptes innerhalb einer föderierten Datenbankumgebung auftretenden Probleme sind offensichtlich. Jede Datenbankoperation, welche die Konsistenz der Daten verletzt, muß kontrolliert werden.

Ein vielversprechender Ansatz zur Konsistenzsicherung wird mit *aktiven Mechanismen* verbunden. Ein Ziel des Projektes **SIGMA**$_{FDB}$ besteht in der Erweiterung des Konzeptes aktiver Mechanismen um eine Anwendbarkeit in föderierten Datenbanken zur systemübergreifenden Konsistenzsicherung [CT95].

Zur Realisierung eines solchen Konzeptes sind zunächst geeignete Erweiterungen bestehender aktiver Konzepte zu finden, um diese datenbankübergreifend anwenden zu können. Als Grundlage dienen *ECA–Regeln*, die wie folgt als globale Integritätsregeln eingesetzt werden können:

Event: potentiell konsistenzverletzendes Ereignis
Condition: globale Bedingung formuliert über dem FDBS-Zustand
Action: globale Aktion(en) zur Erhaltung der FDBS–Konsistenz

Im Zusammenhang mit dieser Beschreibung ergeben sich einige wichtige Fragestellungen bezüglich des Einsatzes von ECA–Regeln in föderierten Umgebungen.

Zunächst einmal gilt es zu untersuchen, welche Ereignistypen in einem FDBS zur Konsistenzsicherung notwendig sind. Denkbar sind Ereignistypen wie z.B. Zeitereignisse, Methodenaufrufe (Operationsaufrufe) oder auch zusammengesetzte Ereignisse. Untersuchungen zur Erkennung von Ereignissen in FDBS bilden einen weiteren Schwerpunkt. Unterstützen CDBS Trigger-Mechanismen, können diese zur Ereigniserkennung ausgenutzt werden, sofern diese in der Lage sind, Ereignisse über die Grenzen des CDBS zu propagieren. Verfügt ein CDBS nicht über Trigger-Funktionalität, ist ein Kapseln der SQL-Anweisungen erforderlich. Das nachträgliche Kapseln von SQL-Anweisungen bestehender Anwendungen steht allerdings ebenso im Widerspruch zur Forderung nach lokaler Autonomie wie das Ergänzen einer Datenbankanwendung um eine aktive Schicht.

Die zweite Frage betrifft die Formulierung von globalen Konsistenzbedingungen. Diese Bedingungen sollen über die Zustände der einzelnen CDBS spezifiziert werden und dabei gleichzeitig mehrere CDBS betreffen können.

Im Aktionsteil sollen neben einfachen Aktionen, wie beispielsweise SQL–Anweisungen an ein bestimmtes CDBS, auch systemübergreifende Aktionen spezifiziert werden können. Dazu ist es erforderlich, eine Menge von Aktionen in unterschiedlichen CDBS auslösen zu können, was wiederum geeignete Synchronisationsmechanismen erfordert.

ECA–Regeln erscheinen aufgrund ihrer Mächtigkeit und Flexibilität ein geeigneter Mechanismus zur Konsistenzsicherung in föderierten Datenbanken zu sein. Offen ist dabei, wie mächtig die zu verwendende Regelsprache sein soll bzw. darf. Zur Realisierung einer systemübergreifenden Konsistenzsicherung in einem FDBS schlagen wir die in Abbildung 2 dargestellte Architektur vor [CT97]. Ein ähnlicher Ansatz wird in [SSW95] verfolgt.

In der Föderierungsebene werden globale Konsistenzbedingungen in Form von globalen ECA–Regeln bereitgestellt. Die Erkennung globaler Ereignisse wird durch den in die föderierte Schicht integrierten aktiven Mechanismus realisiert. Lokale Ereignisse werden dagegen von dem jeweiligen CDBS registriert. Eine große Anzahl kommerziell verfügbarer CDBS verfügen über verwendbare Triggermechanismen. Andere Datenbanksysteme müssen in Form einer Vermittler-Schicht um aktive Mechanismen erst erweitert werden. Die Mechanismen zur Konsistenzsicherung von aktiven CDBS können direkt zur Integritätsüberwachung eingesetzt werden. Aufgabe der Vermittler-Schicht ist es, alle für die Föderierungsebene relevanten Informationen aus den lokalen Transaktionen zu filtern und sie der Föderierungsschicht zur Verfügung zu stellen. Damit bildet die Vermittler-Schicht einen Mechanismus zur Wahrung der semantischen Integrität und stellt dem Föderierungsdienst Recovery-Informationen der CDBS bereit.

3.2 Schematransformation und -integration

Die Schematransformation und -integration stellt einen wesentlichen Schritt beim Entwurf eines föderierten Datenbanksystems dar. Das Ziel der Schematransformation und -integration besteht in der Überwindung der Heteroge-

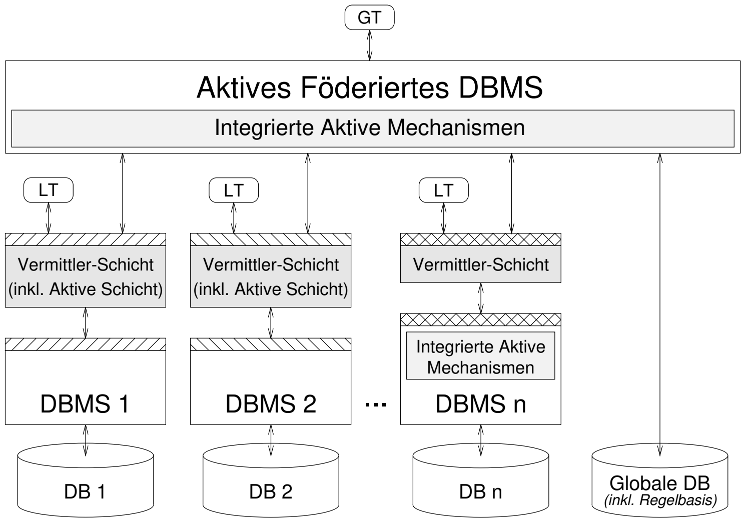

Abb. 2. Aktives föderiertes Datenbanksystem

nität zwischen den verschiedenen Schemata, welche auf unterschiedlichen Datenmodellen basieren. Zur Überwindung der Heterogenität auf Datenmodellebene sind Konzepte erforderlich, die die Konflikte zwischen verschiedenen Schemata auflösen und ein minimales föderiertes Schema erzeugen. Die Anforderung, Konflikte zwischen Datenmodellen aufzulösen, welche unter Benutzung verschiedener Modellierungstechniken entstanden sind, bedingt das Vorhandensein der Schemata in einem einheitlichen Datenmodell. Daher erfolgt vor der eigentlichen Schemaintegration die Transformation der Ursprungsschemata in ein kanonisches Datenmodell.

Das eigens entwickelte *Generic Integration Model* (GIM) [SS96a, SS96b] stellt das benötigte kanonische Datenmodell zur neutralen Beschreibung der auf verschiedenen Datenmodellen basierenden Schemata dar. Dabei wurde im Gegensatz zu den meisten Vorschlägen [BFN94] kein objektorientiertes Datenmodell gewählt. GIM unterstützt statt dessen folgende Basiskonzepte:

- *Einfache Datentypen*: Dieses Konzept ist an die erste Normalform relationaler DBMS angelehnt. Es dient der Definition der Intension einer Klasse. Eine Definition von abgeleiteten (berechneten) Attributen wird nicht unterstützt.
- *Nicht überlappende Klassenextensionen* erzwingen entweder die Äquivalenz oder Disjunktheit zwischen zwei Klassenextensionen. Untermengenbeziehungen oder teilweise Überlappungen zwischen Klassenextensionen werden durch dieses Konzept ausgeschlossen.
- *Objektidentifikatoren* (OID)
- *Binäre Relationen*
- *Integritätsbedingungen*: GIM unterstützt ein dem Standard SQL2 ähnliches Formulieren von Integritätsbedingungen.

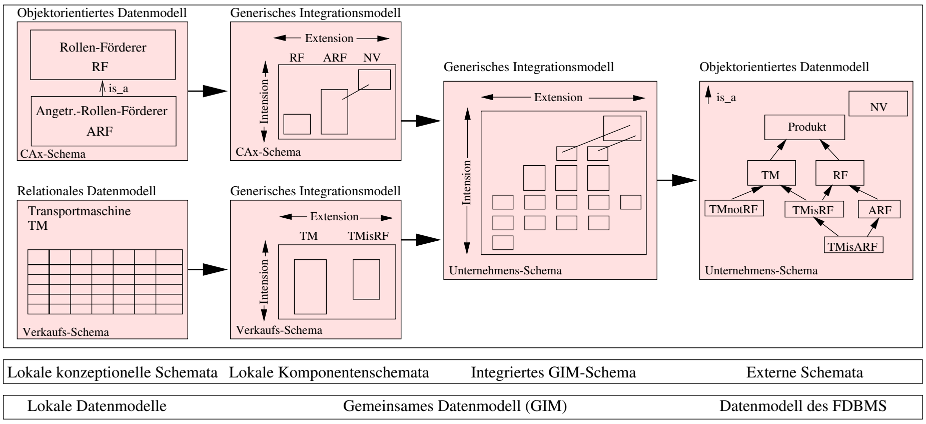

Abb. 3. Beispielszenario für das Generic Integration Model

Aufgrund der semantischen Armut des Datenmodells kann es nicht als Anwendungsdatenmodell dienen. Es ist nur für den Einsatz innerhalb des Integrationsprozesses geplant. Der Zwang zur Transformation der Eingangsschemata nach GIM führt zu normalisierten Eingangsschemata, deren Heterogenität dadurch erheblich eingeschränkt wird. In [SS96a, SS96b] wird gezeigt, wie die Verwendung von GIM zu einer systematischen Integration eingesetzt werden kann. Insbesondere können Algorithmen zur Ableitung objektorientierter externer Schemata auf der Basis eines integrierten GIM-Schemas genutzt werden. Diese Algorithmen erlauben die Ableitung unterschiedlicher externer Schemata, wobei die Forderung nach logischer Datenunabhängigkeit auch von einem FDBS erfüllt werden kann. Abbildung 3 zeigt das Beispielszenario einer Schemaintegration mit GIM.

In diesem Szenario werden ein objektorientiertes und ein relationales Modell föderiert. Dazu werden zunächst beide lokalen konzeptionellen Schemata in ein äquivalentes lokales Komponentenschema transformiert. Die lokalen Komponentenschemata werden bereits durch das generische Integrationsmodell – GIM ausgedrückt. Aus den lokalen Komponentenschemata wird ein minimales föderiertes Schema (integriertes Schema) errechnet, von welchen mehrere externe Schemata abgeleitet werden können.

3.3 Integration von Dateien in eine Datenbankföderation

Ein bislang nur wenig untersuchtes Gebiet ist die Integration von Dateienverbünden in Datenbankföderationen. Dabei verstehen wir unter einem *Dateienverbund* eine Menge von Dateien, die von einer oder von mehreren Applikationen zur persistenten Datenspeicherung und teilweise auch zur Kommunikation genutzt werden. Wir haben im Rahmen von Untersuchungen an Fabrikplanungssystemen den Bedarf für die konzeptionelle Bearbeitung dieses Problems erkannt [HTJ+95]. Dies betrifft vor allem Informationssysteme, die für sehr spezielle Anwendungsfelder entwickelt wurden und die oft in PC-Umgebungen implementiert sind. Trotz der funktionalen Ausrichtung solcher Systeme werden in ihnen oft beachtliche Mengen von Daten verwaltet. Auch die Integration von HTML-Dateien stellt ein großes Potential dar. Der grundsätzlich vorhandene Bedarf nach Integration verschiedener dateibasierter Systeme wird durch die zunehmende Verfügbarkeit von Vernetzungstechniken verstärkt. Die Anwender kommen dabei schnell zu der Erkenntnis, daß das durch die Überwindung der technischen Heterogenität möglich gewordene schnelle Kopieren von Dateien nicht ausreicht. Nun wird die Notwendigkeit der Überwindung der syntaktischen und semantischen Heterogenität offensichtlich, wobei die Autonomie der zu integrierenden Systeme erhalten werden muß.

Als grundlegendes Problem bei der Integration von Dateienverbünden kann man das Fehlen sowohl eines konzeptionellen Schemas auf Basis eines fundierten Datenmodells als auch einer definierten komfortablen Datenbankschnittstelle postulieren. Der Entwurf spezieller Transformatoren, die ein Dateiformat in ein anderes umsetzen, ist oft zwar prinzipiell möglich, erweist sich aber immer wieder als sehr aufwendiger und fehleranfälliger Prozeß. Wir schlagen deshalb ein systematisches Vorgehen vor, das auf einer Klassifikation ähnlicher Dateienverbünde basiert und das für jede Klasse spezifische Analysemethoden und -werkzeuge bietet (vgl. [Höd96]). Für die Schemaableitung stehen *Hintergrundwissen über das Anwendungsszenario und über das lokale Informationssystem*, *Beispieldateien*, *Informationen über Zugriffsoperationen* und teilweise *Quelltexte von Anwendungsprogrammen* zur Verfügung. Der erste Schritt besteht in der Anreicherung eines allgemeinen Wörterbuches um spezifische Beispieldaten und vermutete Metadaten, die bei der Analyse des Hintergrundwissens gewonnen wurden. Unter Zuhilfenahme dieses erweiterten Wörterbuches wird anhand von Beispieldateien und Zugriffsinformationen eine Zuordnung des Dateienverbundes zu einer Klasse vorgenommen. Nun kommen klassenspezifische Analysemethoden und -werkzeuge zum Einsatz, die den Designer beim Entwurf einer Strukturbeschreibung und, im weiteren, eines konzeptionellen Schemas unterstützen. Für die Formulierung der Strukturbeschreibung schlagen wir die Nutzung der Sprache SGML [ISO86] und einer zusätzlichen Typbeschreibung vor. SGML zeichnet sich gegenüber der Nutzung einer Grammatik durch Formalität bei gleichzeitiger Transparenz aus. Bei der Modellierung des konzeptionellen Schemas sollte bereits das kanonische Datenmodell des FDBS benutzt werden. Das abgeleitete Schema kann somit unmittelbar als Komponentenschema dienen, wodurch eine zusätzliche Transformation vermieden wird.

Zusammenfassend sei bemerkt, daß die vorgestellten Ansätze und insbesondere die vorgeschlagene Klassifikation einer weiteren Untersuchung bedürfen. Die entworfenen Methoden und Algorithmen werden in der im folgenden vorgestellten FDBS-Entwurfsumgebung **SIGMA**$_{BENCH}$ prototypisch realisiert und somit validiert.

3.4 SIGMABENCH — Werkzeug zum Entwurf eines FDBMS

Mit der **SIGMA**$_{BENCH}$ wird der Versuch unternommen, einen Prototyp zur Unterstützung des Entwurfs föderierter Datenbanken zu implementieren. Dabei beinhaltet dieses Integrationswerkzeug eine Menge von Programmen, welche die

– Datenmodell- bzw. Dateianalyse,
– Schemaintegration und
– Ableitung externer Schemata

unterstützen bzw. umsetzen. Im Rahmen der Datenmodell- bzw. Dateianalyse werden bestehende lokale Datenmodelle hinsichtlich der enthaltenden Intensionen und Extensionen sowie der Abhängigkeiten zwischen Intensionen beschrieben. Insbesondere die Analyse der Modelle von auf Dateien basierenden Datenhaltungssystemen wird durch diese Komponente unterstützt, indem aus verfügbaren Dateiinformationen SGML–Beschreibungen teilweise automatisch erzeugt werden.

Die in der Analysephase gesammelten Schemainformationen werden in einem nächsten Schritt dem die Schemaintegration unterstützenden Werkzeug übergeben. Dieses Werkzeug untersucht vorliegende Schemata auf intensionale und extensionale Überlappungen sowie auf Namenskonflikte. Grundlage dieses Werkzeugs bildet das in Abschnitt 3.2 vorgestellte Generische Integrationsmodell. Die Schemabeschreibung des aus der Integration resultierenden Datenmodells liegt als objektorientiertes Schema vor. Während der Programmabarbeitung werden intensionale und extensionale Überlappungen visualisiert, und dem Designer der föderierten Datenbank wird eine Unterstützung zur Namensfindung beim Ableiten der externen Schemata angeboten.

4 Zusammenfassung und Ausblick

Im Rahmen dieses Beitrages stellten wir das Projekt **SIGMA**$_{FDB}$ vor. Dazu diskutierten wir aufbauend auf der Vorstellung der Grundlagen föderierter Datenbanken sowohl die Ziele des Projektes als auch erste Lösungsansätze. Insbesondere in den Bereichen systemübergreifende Konsistenzsicherung, Schemaintegration und Einbeziehung von Dateien in Datenbankföderationen können erste Ergebnisse verzeichnet werden, welche sich teilweise in einer prototypisch umgesetzten Entwurfsumgebung für föderierte Datenbanken wiederspiegeln. Ergänzt werden diese Ergebnisse durch Untersuchungen zu Sicherheitsaspekten in Datenbankföderationen und zur Transaktionsverwaltung in föderierten Datenbanken.

Ferner werden wir prüfen, ob die bei der Anwendung von FDBS im Bereich der Fabrikplanung gewonnenen Erkenntnisse sich auf andere Bereiche ausdehnen lassen, in denen redundante und inkonsistente Datenbestände vorhanden sind bzw. der Bedarf nach einer systemübergreifenden Konsistenzsicherung besteht. Beispielhaft seien hier Krankenhausinformationssysteme, Touristiksysteme oder auf verteilten Objektumgebungen aufbauende Systeme genannt.

Zusammenfassend kann festgestellt werden, daß mit FDBS ein vielversprechender Ansatz zur Verwaltung heterogener Datenbestände zur Verfügung steht, welcher jedoch weiterer Forschungsanstrengungen bedarf.

Literatur

[BFN94] R. Busse, P. Fankhauser und E. J. Neuhold. Federated Schemata in ODMG. In J. Eder und L. A. Kalinichenko, Herausgeber, *East/West Database Work-*

shop, Proc. of the 2nd Int. Workshop, Klagenfurt, Austria, September 1994, S. 356–379. Workshops in Computing, Springer-Verlag, London, 1994.

[BLN86] C. Batini, M. Lenzerini und S. B. Navathe. A Comparative Analysis of Methodologies for Database Schema Integration. *ACM Computing Surveys*, 18(4):323–364, Dezember 1986.

[CSGS94] M. Castellanos, F. Saltor und M. Garcia-Solaco. A Canonical Model for the Interoperability Among Object-Oriented and Relational Databases. In M. T. Özsu, U. Dayal und P. Valduriez, Herausgeber, *Distributed Object Management*, S. 309–314. Morgan Kaufmann Publishers, San Mateo, CA, 1994.

[CT95] S. Conrad und C. Türker. Active Integrity Maintenance in Federated Database Systems. Preprint Nr. 9, Fakultät für Informatik, Universität Magdeburg, November 1995.

[CT97] S. Conrad und C. Türker. Unternehmensweite Datenkonsistenz durch Föderierung bestehender Informationssysteme. In *Proc. Wirtschaftsinformatik (WI'97)*, 1997. *To appear.*

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

[Höd96] M. Höding. An Approach to Integration of File Based Systems into Database Federations. In *Proc. of the Int. Workshop on Heterogeneous Information Management, Prague, Czech Republic, November 1996*, 1996. *To appear.*

[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 $\mathbf{SIGMA}_{FDB}$. Preprint Nr. 12, Fakultät für Informatik, Universität Magdeburg, Dezember 1995.

[ISO86] ISO (International Organization for Standardization). *The Standard Generalized Markup Language.* (ISO 8879), 1986.

[RS94] E. Radeke und M. H. Scholl. Federation and Stepwise Reduction of Database Systems. In W. Litwin und T. Risch, Herausgeber, *Applications of Databases, Proc. of the 1st Int. Conf., ADB-94, Vadstena, Sweden, June 1994*, S. 381–399. LNCS 819, Springer-Verlag, Berlin, 1994.

[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.

[SS96a] I. Schmitt und G. Saake. Integration of Inheritance Trees as Part of View Generation for Database Federations. In B. Thalheim, Herausgeber, *Conceptual Modelling — ER'96, Proc. of the 15th Int. Conf., Cottbus, Germany, October 1996*, S. 195–210. LNCS 1157, Springer-Verlag, Berlin, 1996.

[SS96b] 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)*, S. 751–758, 1996.

[SSW95] W. Schaad, H.-J. Schek und G. Weikum. Implementation and Performance of Multi-level Transaction Management in a Multidatabase Environment. In O. Bukhres, T. Özsu und M.-C. Shan, Herausgeber, *RIDE-DOM'95, Proc. of the 5th Int. Workshop on Research Issues in Data Engineering: Distributed Object Management*, S. 108–115. IEEE Computer Society Press, 1995.