Veröffentlicht als: Preprint Nr. 12, Fakultät für Informatik, Universität Magdeburg, Dezember 1995.

Föderierte Datenbanksysteme - Grundlagen und Ziele des Projektes SIGMA$_{\boldsymbol{FDB}}$[1]

Michael Höding Can Türker Sven Janssen Kai-Uwe Sattler
Stefan Conrad Gunter Saake Ingo Schmitt

Otto-von-Guericke-Universität Magdeburg
Institut für Technische Informationssysteme
Postfach 4120, D–39016 Magdeburg
Email: {hoeding|tuerker|janssen|sattler|conrad|saake|schmitt}@iti.cs.uni-magdeburg.de
WWW: http://wwwiti.cs.uni-magdeburg.de/sigma.html
Tel.: ++49/391/67-{12220|18800}
Fax: ++49/391/67-12020

Dezember 1995

[1]Diese Arbeit wurde gefördert vom Land Sachsen-Anhalt under 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*)"

Zusammenfassung

In diesem Bericht stellen wir die wesentlichen Grundlagen und Ziele des seit September 1995 laufenden Projektes **SIGMA**$_{FDB}$, das sich mit der Entwicklung einer Basis-Informationsinfrastruktur als Grundlage integrierter und einheitlicher Datenhaltung für alle Phasen der Fabrikplanung beschäftigt, dar. Unter Berücksichtigung der durch das Anwendungsgebiet Fabrikplanung vorgegebenen Randbedingungen motivieren wir den Einsatz eines föderierten Datenbanksystems als Rahmensystem zur Integration der an der Fabrikplanung beteiligten Software-Werkzeuge einschließlich ihrer lokalen Datenbestände. Wir geben zunächst eine kurze, allgemeine Einführung in das Gebiet föderierter Datenbanksysteme, skizzieren die speziellen Projektziele und stellen für das Projekt relevante Standards und Prototypen föderierter Datenbanksysteme vor. Abschließend diskutieren wir erste Ideen zu den Forschungsschwerpunkten *Schematransformation- und Integration* und *aktive Konsistenzsicherung.*

Schlüsselwörter: föderierte Datenbanksysteme, Schemaintegration, Datenbankentwurf, globale Konsistenzsicherung, ECA-Regeln, aktive Datenbanken, **SIGMA**$_{FDB}$.

Danksagung

An dieser Stelle möchten wir uns bei unseren Partnern aus der Fakultät Maschinenbau, insbesondere bei Olaf Wolter, Carsten Schmidt und Reinhard Fietz, für die hilfreichen Diskussionen bedanken. Unser besonderer Dank für die redaktionelle Unterstützung und für fleißiges Korrekturlesen gilt Frau Andrea Diekmann.

Inhaltsverzeichnis

Kapitel 1

Einleitung

Im Rahmen des Forschungsschwerpunktes "Neue Produktionssysteme/Experimentelle Fabrik" fördert das Land Sachsen-Anhalt das Forschungsprojekt **SIGMA**$_{FDB}$[1], das der Erforschung föderierter Datenbanksysteme als Basistechnologie für die Integration heterogener Softwaresysteme dient. In diesem Bericht stellen wir wichtige Grundlagen und Techniken föderierter Datenbanksysteme (FDBS) vor. Anhand einiger Beispiele veranschaulichen wir die Anwendung von FDBS im Bereich Fabrikplanung. Wichtige einzubeziehende Standards und protoypische Systeme werden kurz diskutiert. Weiterhin präsentieren wir erste Überlegungen zu den Projektschwerpunkten föderierter Datenbankentwurf und föderierte Konsistenzsicherung.

In diesem ersten Kapitel geben wir eine Motivation zur Erforschung und zum Einsatz föderierter Datenbanksysteme. Wir zeigen den Bedarf der Praxis an Integrationslösungen für heterogene Datenhaltungssysteme und die Möglichkeiten föderierter Datenbanksysteme, zur Erfüllung dieses Bedarfes beizutragen. Ferner erläutern wir die Einordnung des Projektes in das interdisziplinäre Forschungsvorhaben "Virtuelle Fabrik". Abschließend folgt die Vorstellung der Kapitelstruktur dieses Berichts.

1.1 Motivation

In den letzten Jahrzehnten haben sich Datenbanksysteme in vielen Bereichen von Wirtschaft, Verwaltung und Wissenschaft durchgesetzt [Bla95]. Insbesondere die Enwicklung relationaler Datenbanksysteme erschloß neue Anwendungsfelder jenseits des traditionellen Datenbankeinsatzes in Banken, Fluggesellschaften, Bibliotheken, etc. Auch in klein- und mittelständischen Unternehmen ist die Nutzung von Informationssystemen auf der Basis moderner Datenbanksysteme ein wichtiger Katalysator des Erfolgs geworden. Neben der Anwendung in Auftragsabwicklung, Rechnungswesen, etc. haben sich in den letzten Jahren weitere technische Informationssysteme wie z.B. CAD- und CAE-Systeme durchgesetzt [Pau93, PWPH95]. Auch sie verwalten produkt- und unternehmensspezifische Daten, zum

[1] **SIGMA**$_{FDB}$ (**S**_chema_ **I**_ntegration and_ **G**_lobal integrity_ **M**_aintenance_ **A**_pproach_ _for_ **F**_ederated_ **D**_ata_ **B**_ases_)

Teil bereits unter Nutzung von Datenbanksystemen [PWPH94]. So ist in vielen Unternehmen eine Landschaft heterogener Datenbank- und Datenhaltungssysteme entstanden.

Der breite Einsatz von Vernetzungstechnologien zur unternehmensweiten und zum Teil unternehmensübergreifenden Kopplung der Rechnersysteme erlaubt prinzipiell die integrierte Nutzung aller im Verbund existierender Informationen [Rah94]. Aus dieser Möglichkeit und dem akuten Bedarf nach schnell verfügbaren, aktuellen und korrekten Informationen, die für das dynamische Agieren der Unternehmen im Markt entscheidend sind, resultiert die Forderung nach Diensten für den systemübergreifenden Zugriff auf die verschiedenen Datenbestände [EHSS91]. Wesentlich ist dabei die Schaffung und Erhaltung der Konsistenz für den gesamten Datenbestand. Dabei sind Probleme zu lösen, die durch die redundante Datenhaltung in den Einzelsystemen bedingt sind.

Ähnliche Forderungen bestehen auch im Bereich der integrierten Nutzung autonomer Werkzeuge zur Lösung von Spezialaufgaben [BG95, Zie95, ZWB95]. Diese Werkzeuge verfügen zumeist über eigene Datenhaltungssysteme, zum Teil auf Basis von Dateisystemen. Die Kopplung der Werkzeuge erfolgt dabei über den Im- und Export sowie die Transformation von Dateien. Auch hierbei treten Probleme durch redundante Datenbestände auf. Verbesserungspotentiale liegen in der Unterstützung von synchroner, interaktiver, kooperativer und konkurrierender Arbeit [Str93]. Damit wird es möglich, Produktionszykluszeiten zu verkürzen bzw. die Produktqualität zu erhöhen.

Die meisten der auftretenden Probleme resultieren aus den verschiedenen Arten von Heterogenität, wie

- Heterogenität von Hardware und Betriebssystemen,
- Heterogenität der zugrunde liegenden Datenmodelle,
- Heterogenität der Datenhaltungs- bzw. Datenbankmanagementsysteme (unterschiedlicher Funktionsumfang, unterschiedliche Hersteller, verschiedene Versionen)
- Heterogenität der Schemata und
- Heterogenität der Daten selbst.

Ein naheliegender Ansatz ist deshalb die Überwindung der Heterogenität durch den Einsatz eines zentralen Datenbanksystems, das alle Daten einheitlich und zentral verwaltet [MSJ94]. Versuche, unternehmensweite Datenmodelle zu entwickeln und einzusetzen, führten zu sehr mächtigen Systemen, die teilweise schwer handhabbar, unübersichtlich und langsam sind. Außerdem gibt es kein Datenbankmanagementsystem, das für alle Einsatzfälle gleichermaßen geeignet ist. Ein weiteres Problem ist der enorme Aufwand bei der Einführung eines zentralen Systems. Bereits existierende Anwendungen, die zumeist eine große Menge Know-how des Unternehmens darstellen, müssen für das neue zentrale System umprogrammiert werden. Dies ist teilweise nicht möglich, da in einigen Fällen kein Quellcode zu Verfügung steht. Ein weiteres Problem ist die Migration der “ererbten” Datenbestände. Die Nachteile des Einsatzes eines zentralen Systems lassen die großen Risiken und den erheblichen Aufwand eines solchen Umstellungsschrittes erkennen.

Eine andere vielversprechende Möglichkeit zur Überwindung der Heterogenität ist die Nutzung föderierter Datenbanksysteme [HM85, SL90, LMR90, BHP92]. Bestehende (heterogene) Datenbank- und Datenhaltungssysteme werden zu einem umfassenden System föderiert, wobei die Einzelsysteme ihre Autonomie weitgehend bewahren. Dies hat den Vorteil, daß bestehende Anwendungen unverändert und weitgehend ohne Performance-Verlust weiterlaufen können, während neuen Anwendungen die Möglichkeit geboten wird, über eine einheitliche Schnittstelle auf die föderierten Datenbestände zuzugreifen. Damit wird die Einführung neuer bzw. eine sanfte Migration bestehender Applikationen möglich, womit das Fehlschlagrisiko beträchtlich sinkt.

In Erweiterung der Dreiebenen-Architekter nach ANSI/SPARC [DAF86] wird die Einführung einer Föderierungsschicht vorgeschlagen [SL90]. Diese Föderierungsschicht wird durch einen Föderierungsdienst, eine spezielle Middleware, realisiert. Der Einsatz eines solchen Föderierungsdienstes eröffnet qualitativ neue Möglichkeiten für den Datenbankeinsatz.

- *Interoperabilität durch Integration*
  Wie bereits oben beschrieben, ist ein vorrangiges Ziel die Integration der lokalen, autonomen Datenbank- und Datenhaltungssysteme zum Zweck des systemübergreifenden, einheitlichen, transparenten sowie konsistenten lesenden und schreibenden Zugriffs [BG95].

- *Sanfte Migration*
  Durch den Einsatz eines Föderierungsdienstes wird die sanfte Migration, d.h. die Umstellung von einem veralteten Datenbanksystem auf ein neues modernes Datenbanksystem unterstützt [RS94]. Die Migrationsproblematik ist eines der akutesten Probleme bei der Datenbankanwendung in der Praxis [BS95].

- *Datenbankunabhängigkeit*
  Entwickler und Anbieter von Applikationen können durch den Einsatz einer Middleware, z.B. realisiert durch einen Föderierungsdienst, Datenbankunabhängigkeit offerieren [KBB+95]. Dadurch ist es möglich, bei der Einführung solcher Applikationen bereits vorhandene Datenbankmanagementsysteme zu nutzen, wodurch Investitionsmittel besser ausgenutzt werden können.

Diese und weitere Aspekte zeichnen den Einsatz von Föderierungstechniken gegenüber anderen Ansätzen aus und lassen sie als Basistechnologie für die Koordination, Kooperation und Kommunikation durch Datenkopplung geeignet erscheinen.

Die Konzepte für Entwicklung, Einführung und Betrieb föderierter Datenbanksysteme sind Gegenstand der aktuellen Datenbankforschung. Wichtige Aspekte des Forschungsgebietes sind Methoden zur Schematransformation und -integration sowie zur Konsistenzsicherung in föderierten Systemen. Aus diesem Grund sind sie Schwerpunkte des Forschungsprojektes **SIGMA**$_{FDB}$.

Durch Schematransformation wird die syntaktische Heterogenität, durch Schemaintegration die semantische Heterogenität der lokalen Schemata überwunden [SL90, SCG93]. Dabei sind eine Reihe von auftretenden Konflikten zu erkennen und zu lösen [SP91, Dup94, SPD92]. Trotz zahlreicher Veröffentlichungen zur Föderierungsthematik und speziell zur Lösung einzelner Konflikte existiert noch keine geschlossene Methodik, die die Integration von lokalen Schemata zu einem föderierten Schema beschreibt, das den Forderungen von [BLN86] nach Korrektheit, Minimalität und Verständlichkeit entspricht. Auch der wesentliche Aspekt der systemübergreifenden Konsistenzsicherung in bzw. durch föderierte Datenbanksysteme, der besonders für die Kontrolle von Redundanzen wichtig ist, wurde in der Literatur bislang kaum diskutiert. Erste Ansätze sind in [CT95, TC95] skizziert.

Die Aufarbeitung dieser Defizite ist Gegenstand des Projektes $\mathbf{SIGMA}_{FDB}$. Dabei gewonnene Erkenntnisse und abgeleitete Methoden sollen prototypisch realisiert und anhand von Beispielen validiert werden. Dabei soll eines der derzeit auf den Markt drängenden kommerziellen Föderierungssysteme, z.B. OpenDM/Efendi [Rad94, BK95], IRO-DB [BFHK94] oder ZOO-IFI [Här94] zum Einsatz kommen.

1.2 Einordnung des Projektes

Das Projekt ordnet sich in den Forschungsschwerpunkt *Experimentelle Fabrik - Neue Produktionssysteme* des Landes Sachsen-Anhalt ein. In diesem Rahmen haben sich Wissenschaftler aus Informatik und Maschinenbau zu interdisziplinärer Zusammenarbeit gefunden. Die Zusammenarbeit basiert auf dem gemeinsamen Thema *Virtuell-Explorative Fabrikplanung* und dient der Erforschung neuer Wege und Methoden für die integrierte Unterstützung aller Phasen im Lebenszyklus moderner Fabriken durch den Einsatz modernster Informationstechnologie.

In diesem Sinne ist es Aufgabe des Projektes $\mathbf{SIGMA}_{FDB}$, Grundlagen für die Informations- und Kommunikationsinfrastruktur der *Virtuellen Fabrik* zu schaffen. Dabei geht es sowohl um die Integration von Werkzeugen zur Planung und Entwicklung, als auch um Aspekte der informationstechnischen Unterstützung des Betriebes. Auf Basis der bereits oben aufgeführten Argumente, verbunden mit den nur begrenzt zur Verfügung stehenden personellen und materiellen Kapazitäten, ist der Einsatz von Föderierungstechniken zu empfehlen. Damit wird in Spezial-Applikationen enthaltenes Know-how im Sinne der Investitionssicherung, bei gleichzeitiger Flexibilität des Gesamtsystems, erhalten.

Die Beispielszenarien, die wir in diesem Bericht vorstellen und diskutieren, sind u.a. durch die interdisziplinäre Zusammenarbeit mit Projektpartnern aus dem Maschinenbau motiviert. Sie dienen in erster Linie zur Verdeutlichung allgemeiner Probleme bei der Entwicklung und dem Betrieb föderierter Datenbankssysteme. Darüber hinaus sollen sie aber auch helfen, eine Basis für die Verständigung von Informatikern und Maschinenbauern bei der gemeinsamen Arbeit zu schaffen, denn auch in diesem Kontext geht es um die Integration zweier heterogener Begriffs- und Verständniswelten.

1.3 Vorgehen und Kapitelstruktur

Ziel dieses Berichtes ist es, Grundlagen des Forschungsprojektes **SIGMA**$_{FDB}$ aufzuarbeiten und erste Ideen zu den Forschungsgebieten Schematransformation und -integration sowie Konsistenzsicherung in FDBS zu präsentieren.

Nach der in diesem Kapitel gegeben Motivation werden im Kapitel 2 wichtige Grundbegriffe zum Forschungsgebiet föderierte Datenbanksysteme vorgestellt. Dabei geht es um die Einordnung in das Gebiet der Multidatenbanksysteme, Architekturaspekte, die Entwicklung und den Betrieb föderierter Datenbanksysteme. In Kapitel 3 wird eine Anforderungsanalyse für die im Rahmen des Projektes zu untersuchenden Aufgabenstellungen durchgeführt. Dabei werden sowohl allgemeine Anforderungen an föderierte Datenbanksysteme als auch projekttypische Anforderungen diskutiert. Ein Überblick über wichtige Standards, Technologien und Systeme wird im Kapitel 4 gegeben. Dabei wird eine Bewertung hinsichtlich der Relevanz für das Projekt **SIGMA**$_{FDB}$ vorgenommen. In Kapitel 5 wird näher auf die für die Entwicklung föderierter Datenbanksysteme wichtigen Phasen der Schematransformation und -integration eingegangen. Vor- und Nachteile verschiedener kanonischer Datenmodelle und Integrationstrategien werden diskutiert, eine Klassifikation auftretender Schemakonflikte wird vorgestellt. Gegenstand von Kapitel 6 ist die systemübergreifende Konsistenzsicherung in föderierten Systemen. Dabei werden aktive Mechanismen als aktuelles und vielversprechendes Lösungsprinzip diskutiert. In Kapitel 7 werden wesentliche Ergebnisse und Erkenntnisse des Berichtes zusammengefaßt. Abschließend folgt ein Ausblick auf zukünftige Arbeiten.

Kapitel 2

Grundlagen

Gegenstand dieses Kapitels sind die für das Gebiet föderierter Datenbanksysteme (FDBS) wichtigen Grundlagen und Begriffe. Nachdem wir eingangs eine Begriffsbestimmung und Einordnung der föderierten Datenbanksysteme in den Kontext der verteilten Datenbanksysteme vornehmen, wird genauer auf die Architektur von FDBS eingegangen. Dabei wird die 5-Ebenen-Referenz-Architektur [SL90] für föderierte Datenbanksysteme vorgestellt. Im weiteren werden spezielle Aspekte des Softwarelebenszyklus für FDBS diskutiert.

2.1 Begriffsbestimmung und Einordnung

Ein föderiertes Datenbanksystem besteht aus dem föderierten Datenbankmanagementsystem und aus einem oder mehreren *Komponentendatenbanksystemen* (CDBS). Das föderierte Datenbankmanagementsystem wird oft auch als Föderierungsdienst oder Föderierungsschicht bezeichnet [HM85, KBB$^+$95]. Das FDBS dient dazu, logisch und möglicherweise physisch verteilte heterogene Datenbanksysteme bei weitgehender Erhaltung der lokalen Autonomie der CDBS zu föderieren.

Die Komponentendatenbanksysteme können dabei hinsichtlich der Laufzeitumgebung, des Datenmodells und der Datenbankschemata heterogen sein [Rah94]. Es ist möglich, daß ein CDBS seinerseits wieder als föderiertes Datenbanksystem realisiert ist. 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 föderierte Datenbankmanagementsystem operiert ebenfalls auf den lokalen Schnittstellen der einzelnen CDBS. Bei vollständiger Wahrung der Autonomie der CDBS verhält sich der Föderierungsdienst wie eine lokale Applikation. Aufgabe des Föderierungsdienstes ist die Überwindung der Heterogenität. Es werden ein oder mehrere föderierte Schemata für den einheitlichen, transparenten und integrierenden Zugriff auf alle föderierten Daten realisiert. Die Darstellung des föderierten Schemas basiert auf einem einheitlichen Datenmodell, dem sogenannten *kanonischen Datenmodell*. Über externe Schnittstellen können globale Applikationen die Dienste des föderierten Datenbanksystems nutzen [Hsi92a, Hsi92b].

Der Einsatz verteilter Datenbankssysteme ist oft in der geografischen bzw. aufga-

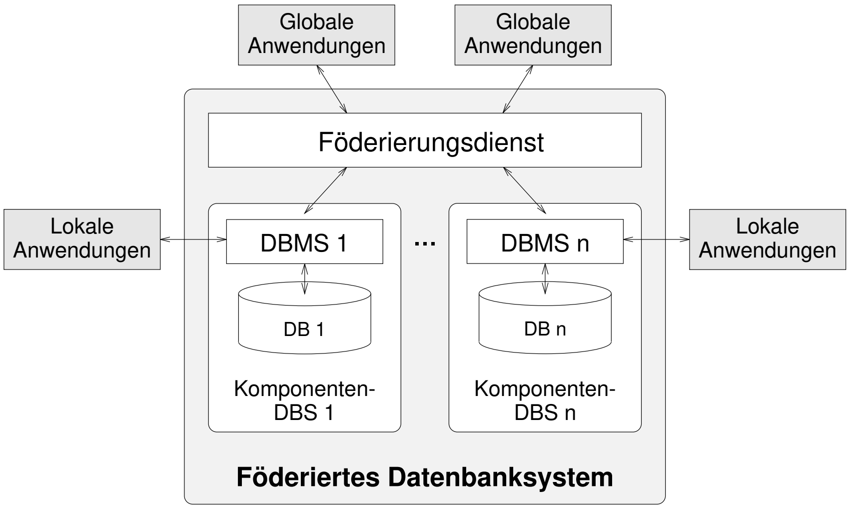

Abbildung 2.1: Komponenten eines Föderierten Datenbanksystem

benspezifischen Trennung der Systeme begründet. Während geografisch getrennte Realisierungen mittels *homogener verteilter Datenbanksysteme* möglich sind, führt die aufgabenspezifische Trennung fast zwangsläufig zum Einsatz *heterogener Datenbanksysteme*. Zur Kopplung solcher heterogener DBS wird neben dem Einsatz von föderierten Datenbankmanagementsystemen auch die Verwendung verteilter Transaktionssysteme diskutiert [Pu87, MKN90, MRB+92]. Eine Übersicht über das Lösungsspektrum zur Unterstützung der Interoperabilität auf heterogenen DBS gibt [Rah94]. In Abbildung 2.2

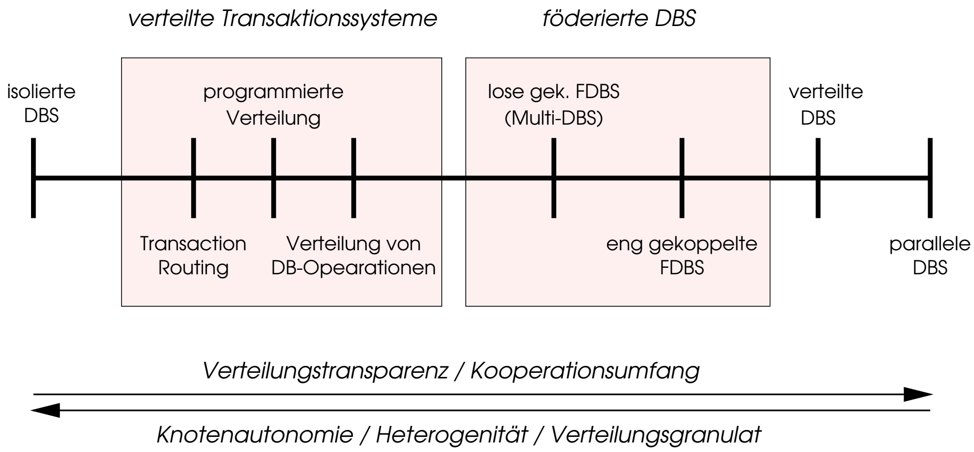

Abbildung 2.2: Lösungsspektrum zur Unterstützung heterogener DBS nach [Rah94]

werden die konträren Aspekte *Verteilungstransparenz/Kooperationsumfang* und *Knotenautonomie/Heterogenität/Verteilungsgranulat* verdeutlicht. Bei föderierten Datenbanksystemen wird in lose [CGW94] bzw. eng [Kac94] gekoppelte FDBS unterschieden. Bei lose gekoppelten Systemen greifen Applikationen mittels einer speziellen Multidatenbanksprache auf die verschiedenen heterogenen Datenbanksysteme zu, d.h. jede Anwendung schafft sich ein eigenes föderiertes Schema. In eng gekoppelten FDBS wird durch den Föderierungsdienst ein globales föderiertes Schema realisiert. Auf dieses Schema bzw. auf davon abgeleitete externe Schemata greifen alle Applikationen zu, die das FDBS nutzen.

Im Rahmen des Projektes $\mathbf{SIGMA}_{FDB}$ konzentrieren wir uns auf eng gekoppelte FDBS, da nur durch die enge Kopplung mittels eines globalen föderierten Schemas die Realisierung globaler Konsistenzsicherungsmechanismen möglich ist. Gerade für das zu untersuchende Gebiet des konkurrierenden und kooperierenden Arbeitens mit integrierten Werkzeugen ist Konsistenzsicherung von wesentlicher Bedeutung.

2.2 Architektur föderierter Datenbanksysteme

Ein *föderiertes Datenbanksystem* (FDBS) besteht aus einer Menge *autonomer heterogener Komponentendatenbanksysteme* (CDBS), die über eine flexible Föderierungsschicht gekoppelt sind. Die wesentliche Aufgabe dieser erweiterbaren Föderierungsschicht besteht darin, eine globale homogene Schnittstelle für den einheitlichen und transparenten Zugriff auf föderierte Daten bereitzustellen. 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 wollen wir im folgenden genauer darauf eingehen. Wir setzen dabei voraus, daß für die Föderierungsschicht ein geeignetes kanonisches Datenmodell existiert.

Die konzeptionellen Schemata (entsprechend der 3-Schema-Ebenen-Architektur nach [DAF86]) der an der Föderierung beteiligten Komponentendatenbanken (CDB) bilden die lokalen Schemata des FDBS. Sie basieren auf dem Datenmodell des jeweiligen Datenbankmanagemensystems. Die lokalen Schemata werden in das kanonische Datenmodell überführt. Dadurch wird die Heterogenität bezüglich des genutzten Datenmodells überwunden. Es enstehen die Komponentenschemata. Durch eine Filterung der an der Föderierung 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 lokalen Systeme unterstützt werden.

Wichtig ist die Einbeziehung von auf Dateien basierenden Datenhaltungssystemen. Dafür ist eine nachträgliche Modellierung der in den Dateien enthaltenen Informationen

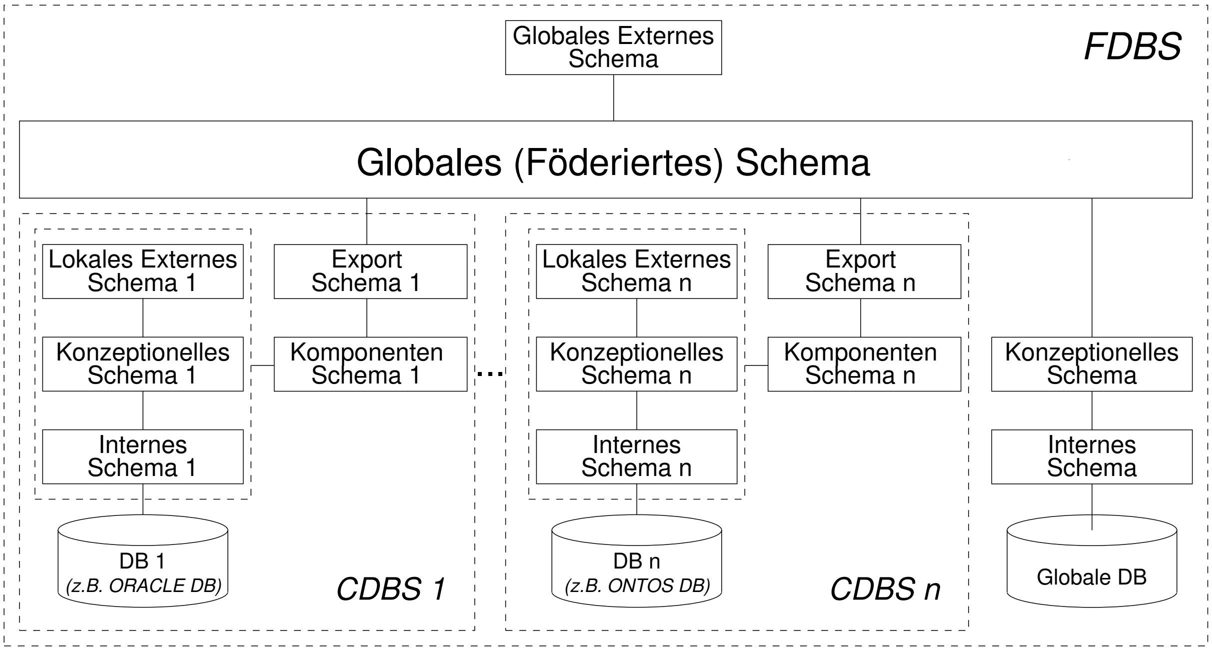

Abbildung 2.3: Schema-Architektur föderierter Datenbanksysteme nach [SL90]

nötig. Die Struktur der Datenabspeicherung kann für diese Systeme als implizites Datenmodell des Datenhaltungssystems aufgefaßt werden. Die nachträgliche Modellierung kann dabei als Transformation in das kanonische Datenmodell aufgefaßt werden. Dieser Vorgang ist sehr aufwendig im Verhältnis zur Transformation zwischen den in etablierten DBS zum Einsatz kommenden Datenmodellen, für die zumeist theoretisch fundierte und praxiserprobte Methoden und Werkzeuge existieren.

Eine wichtiges Kriterium für die Anwendbarkeit von FDBS ist die Erweiterbarkeit [SW91]. Dabei ist vor allem die Unterstützung verschiedener Datenmodelle für CDB, aber auch für externe Schemata von besonderer Bedeutung. In [BEK+94, BK95] wird dabei das im FDBS *Efendi* realisierte Adapterkonzept vorgestellt und diskutiert. Der Föderierungsdienst besteht in diesem System aus einem Kern, Datenbankadaptern und externen Adaptern (siehe Abbildung 2.4). Der Kern nimmt auf Basis des kanonischen Datenmodells die Filterung der Exportschemata und ihre Integration zum föderierten Schema sowie die Datenintegration vor. An den Kern ist eine flexible Anzahl heterogener Datenbankmanagementsysteme bzw. Datenhaltungssysteme über eine variable Menge von Datenbankadaptern ankoppelbar. Diese speziellen Adapter übernehmen die Datenmodelltransformation zur Überwindung der Datenmodellheterogenität für jedes zu unterstützende Datenmodell. Zur Ableitung externer Sichten und zur Erzeugung von Datenmodellheterogenität werden in dieser Architektur externe Adapter eingesetzt. Ein ähnlicher Ansatz wird auch im Rahmen des Projektes FIHD [Jan95] verfolgt.

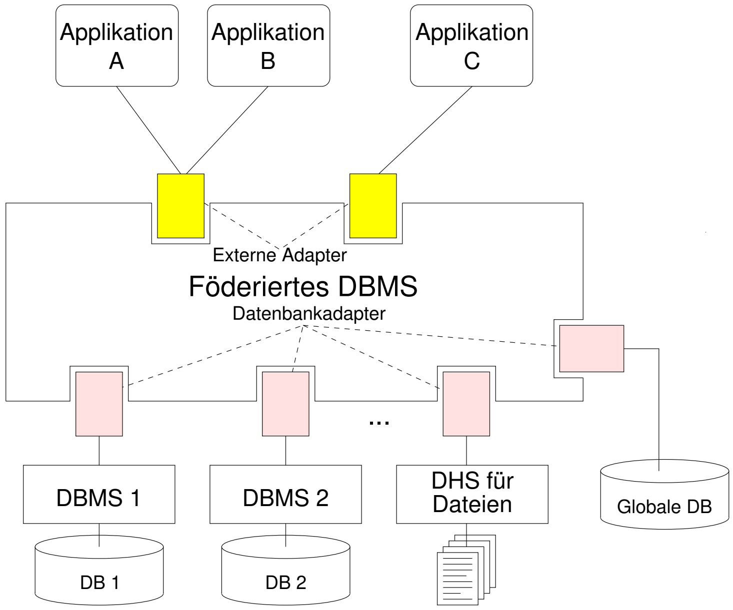

Abbildung 2.4: Realisierung von FDBS mittels Adaptertechnik in Anlehnung an [Rad94]

2.3 Entwicklung und Betrieb von FDBS

Im folgenden wollen wir kurz die Aktivitäten diskutieren, die für die Einführung und den Betrieb eines föderierten Datenbanksystems charakteristisch sind. Sie sind besonders deshalb interessant, weil sich hierbei die Lebenszyklen der Komponentendatenbanksyteme mit dem des föderierten Systems überlappen. Das Phasenmodell des Datenbankentwurfs [Saa93, HS95] muß aus diesem Grunde modifiziert werden (vgl. Abbildung 2.5).

Wir setzen die Existenz eines Föderierungsdienstes voraus und gehen ferner davon aus, daß mehrere heterogene Datenbanken existieren, die ganz oder teilweise zu integrieren sind. Für die Phase der *Anforderungsanalyse* sind die aus dem Datenbankentwurf bekannten Methoden anwendbar. Für einige typische Föderierungsszenarien lassen sich weitere Rahmenbedingungen formulieren. Derartige Anforderungen werden in Kapitel 3 vorgestellt.

Auf Basis der Anforderungsanalyse wird während des *konzeptionellen Entwurfs* ein erstes konzeptionelles Schema abgeleitet (Top-Down). Es ist sinnvoll, dieses konzeptionelle Schema mittels des kanonischen Datenmodells zu beschreiben. In diesem Prozeß sollten Erkennnisse aus der parallel durchzuführenden Ableitung der Exportschemata der CDBS einfließen. Die Bereitstellung der Exportschemata (Bottom-Up) kann dabei entweder durch direkte Übernahme und Filterung der konzeptionellen Schemata der CDBS oder durch die Transformation der lokalen Schmata der CDBS in das kanonische Datenmodell (Reengineering) erfolgen. Die entstandenen Schemata sind zum (konzeptionellen) föderiertem Schema zu integrieren. Dabei auftretende Konflikte sind nicht immer eindeutig lösbar [Sch95, Hil95]. Deshalb sind für spezielle Anwendungen die Auswirkungen zu schließender Kompromisse zu untersuchen und entsprechende Strategien festzulegen. Das auf Grundlage der Anforderungsanalyse gebildete Schema besitzt die meisten Frei-

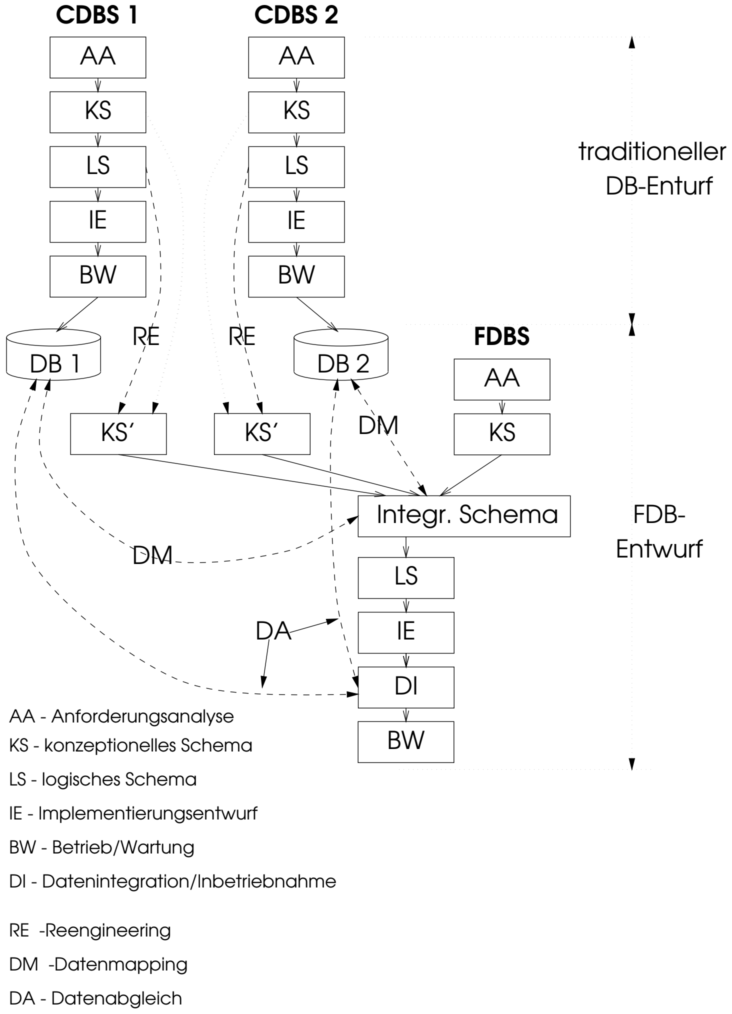

Abbildung 2.5: Phasenmodell des FDB-Entwurfs

heitsgrade hinsichtlich Schemaanpassungen, da es über keine ererbten Daten verfügt. Auf Basis der Exportschemata sind die Abbildungsfunktionen für die Abbildung der Daten zwischen dem föderierten Schema und den lokalen Schemata zu bestimmen.

Die Phase der *Inbetriebnahme* eines FDBS ist durch die oft notwendige *Datenintegration* der durch die lokalen DBSe gehaltenen Daten bestimmt. Wichtig ist vor allem die Erkennung von Objekten der CDB, die dasselbe Realweltobjekt beschreiben, und die Etablierung einer 'same'-Beziehung für diese Objekte [SP91, SPD92]. Es treten in der Regel Datenkonflikte auf, deren Erkennung zum Teil nur sehr schwer möglich ist. Es sind Methoden und Strategien zur Aufdeckung und Behandlung von Redundanzen und Inkonsistenzen anzuwenden [Klu95].

Auch in der Phase des *Betriebs* spielt die Datenintegration eine wichtige Rolle. Die Erhaltung der globalen Konsistenz und die Kontrolle der Redundanz sind wichtige Aufgaben des Föderierungsdienstes. Dabei sind Abstriche bei der Autonomie der CDBS oder der Funktionalität des FDBS hinzunehmen.

Ein weiterer interessanter Aspekt ist die Evolution föderierter Datenbanksysteme. Wir wollen hier nur kurz einige Punkte nennen, die bei der Untersuchung föderierter Datenbankstrukturen von Interesse sind. Die Aufnahme neuer CDB in die Föderation muß gewährleistet sein. Ebenso muß des föderierte Datenbanksystem flexibel auf Schemaänderungen in den CDBS reagieren können. Neben der Datenintegration sollte das FDBS auch die Migration von Daten zwischen den CDBS unterstützen [RS95], um so die sanfte Migration von Applikationen [BS95] zu ermöglichen. Erfolgt die Modifikation des föderierten Schemas, kann dies zur Anpassung der lokalen Schemata führen (Reengineering). Die dabei mögliche Angleichung der lokalen Schemata (im Sinne einer Homogenisierung) bedingt die Verletzung der Entwurfsautonomie der CDBS.

2.4 Zusammenfassung

In diesem Kapitel haben wir wesentliche Grundlagen zum Gebiet föderierte Datenbanken kurz präsentiert. Wir haben eine Einordnung der FDBS in den Bereich der Multidatenbanken vorgenommen, die Architekturkonzepte für FDBS und den Entwurfsprozeß diskutiert. Detaillierte Informationen sind der aufgeführten Literatur, insbesondere [HM85, SL90, BLN86, Hsi92a, Hsi92b, BHP92, Kac94] zu entnehmen.

Kapitel 3

Problem- und Anforderungsanalyse

Nachdem im vorangegangenem Kapitel Grundlagen föderierter Datenbanksysteme vorgestellt wurden, sollen in diesem Kapitel Probleme und Anforderungen an ein FDBS aus Sicht des Projektes **SIGMA**$_{FDB}$ diskutiert werden. Nach einigen einführenden Bemerkungen werden wir anhand von Beispielszenarien, die in dem für das Projekt **SIGMA**$_{FDB}$ relevanten Anwendungsbereichen Fabrik- und Anlagenplanung angesiedelt sind, anwendungsspezifische Probleme des FDBS-Einsatzes betrachten. Abschließend werden die abgeleiteten Anforderungen kurz zusammengefaßt.

3.1 Projektziele

Ein wesentliches Ziel des Projektes ist es, Grundlagen zur Schaffung einer zukunftsorientierten Unternehmens-, Informations- und Kommunikationsinfrastruktur zu untersuchen. Eine solche Infrastruktur ist für die Wettbewerbsfähigkeit von Unternehmen von entscheidender Bedeutung. Die Forderung nach kürzeren Innovationszyklen und qualitativ höherwertigen Produkten bedarf der Umsetzung sowie der informationstechnischen Unterstützung neuer Methoden des integrierten Arbeitens [Zie95]. Dazu gilt es, die Datenbestände der im Unternehmen vorhandenen Softwaresysteme einheitlich zu verwalten, konsistent zu halten und weiterhin zusätzliche Funktionalität zu offerieren. Bei der Einführung dieser Infrastruktur darf es aber nicht zu anwendungsfremden Restriktionen für die lokalen Nutzer kommen. Daraus resultiert die Forderung nach Gewährleistung der Autonomie der zu integrierenden Einzelsysteme. Ein geeignetes Mittel zur Lösung der damit verbundenen Probleme liegt im Einsatz föderierter Datenbanksysteme, die Integrationsmöglichkeiten bei gleichzeitiger Wahrung der lokalen Autonomie bieten. FDBS sind somit eine geeignete Basistechnologie für die Schaffung einer Informations- und Kommunikationsinfrastruktur.

Ein scheinbares Risiko für den Erfolg eines solchen Projektvorhabens mag in der Komplexität der Problemstellung liegen. Wir wollen uns deshalb auf die Untersuchung von Szenarien aus dem Bereich Fabrikplanung [Agg90] beschränken. Ziel der Fabrikplanung ist die optimierte Gestaltung der Fabrikanlage. Während eines mehrstufigen, iterativen Gestaltungsprozesses werden eine Reihe von speziellen Systemen (CAx, PPS, etc.) bzw. ihre

Daten genutzt [ZWB95]. Es existieren heterogene Datenhaltungssysteme mit heterogenen Datenbeständen, die den Einsatz föderierter Datenbanksysteme sinnvoll erscheinen lassen. Die bei der Erforschung dieser Problemstellungen gewonnenen Erkenntnisse können im Rahmen der Projektkooperation mit Fabrikplanern und Logistikern überprüft und abgeglichen werden.

3.2 Bespielszenarien

In Gesprächen mit Fabrikplanern und bei einer ersten Analyse vorhandener Softwaresysteme haben sich die typischen Föderierungsszenarien Integration und Migration herauskristallisiert. Wir wollen beide Szenarien anhand eines Beispiels erläutern und dabei auf spezielle Anforderungen eingehen.

3.2.1 Beispiel 1: Integriertes Fabrikplanungssystem

Wie bereits in den vorhergehenden Kapiteln beschrieben, ist eine zentrale Aufgabe föderierter Datenbanksysteme die Unterstützung des einheitlichen, transparenten und integrierten Zugriffs auf die Daten mehrerer heterogener Datenhaltungssysteme. Ein typischer Anwendungsfall ist die integrierte Nutzung von CAx-Werkzeugen für die Fabrikplanung. Dabei kommen Spezialapplikationen zum Einsatz, die verschiedene Aspekte einer gemeinsamen Miniwelt bearbeiten. Der Planungsprozeß wird in [ZWB95] als Stufenprozeß mit mehreren Iterationen zur schrittweisen Optimierung des Fabrikmodells charakterisiert. Durch das IAF[1] wird das integrierte Fabrikplanungssystem LOGAP (*Log*istik-*A*rbeits*p*latz) entwickelt und eingesetzt [FS94, FS96]. Das System dient der Ermittlung einer optimierten Maschinenaufstellung für eine Produktionshalle bei Neu- oder Umplanungen. Der LOGAP integriert ein Layoutplanungssystem, ein Materialflußdimensionierungstool, ein Simulations- und Animationssystem sowie ein CAD-System. Eingangsdaten sind u.a. die *Grundrisse* der Produktionshalle und die der Maschinen, sogenannte *Sympathien* zwischen Maschinen, die Abhängigkeiten beschreiben, sowie *Arbeitspläne* (z.B. aus PPS-Systemen), die den Produktionsfluß dokumentieren. Alle zum Einsatz kommenden Systeme verfügen über eigene, auf Dateien basierende, Datenhaltungskomponenten. Die Kopplung der Systeme basiert auf dem Austausch von Informationen mittels Dateitransfer. Der LOGAP umfaßt derzeit folgende Softwaresysteme:

- Mit dem CAD-System AUTOCAD werden die Grundrisse der Halle und der Maschinen modelliert.
- Das Layoutplanungssystem LAPLAS ermittelt anhand dieser Grundrisse, von Kostenparametern und von Sympathien eine initiale Maschinenaufstellung.
- Dieser Aufstellungsplan wird mittels des Programms zur Materialflußdimensionierung (DUMAS) unter Berücksichtigung von Arbeitsplänen rechnerisch überprüft.

[1]Institut für Arbeitswissenschaften und Fabrikautomatisierung der Otto-von-Guericke Universität Magdeburg

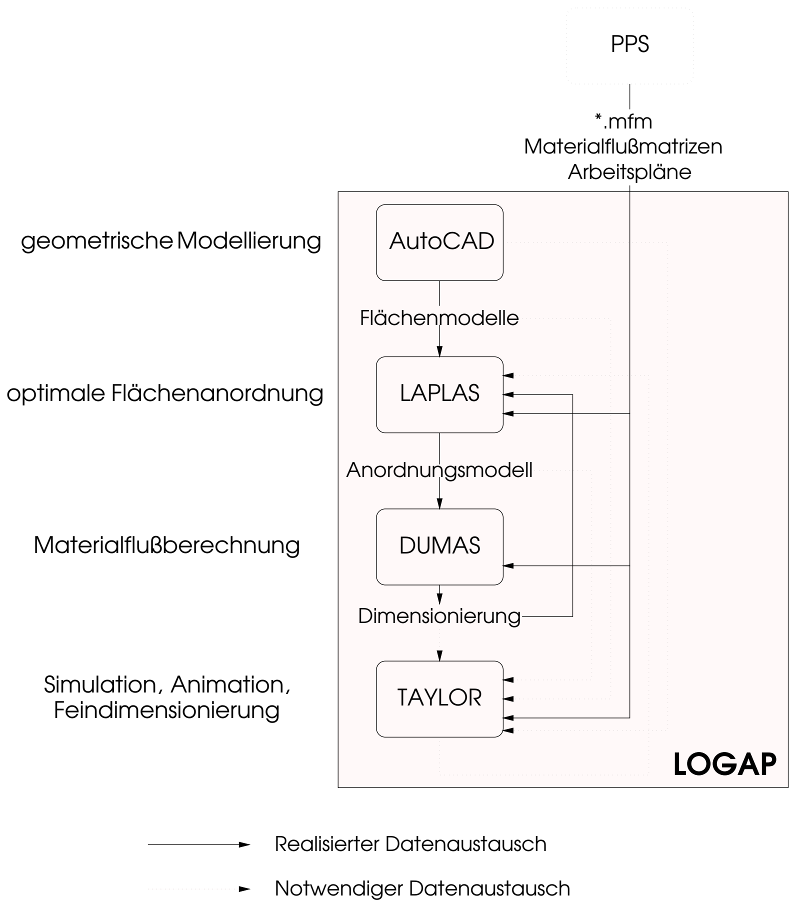

Abbildung 3.1: Datenaustausch zwischen den LOGAP-Komponentensystemen

Es erfolgt eine Dimensionierung der ver- und entsorgenden Materialflüsse, der Produktionspuffer sowie die Berechnung von Grenzleistungen und Warteschlangen.

- Durch das Simulations- und Animationssystem TAYLOR kann das Fabrikmodell simulativ überprüft und der Produktionsprozeß durch eine Animation anschaulich dargestellt werden. Zur Animation muß die Geometrie der Szene nachträglich modelliert werden.

Die Integration der Werkzeuge ist im LOGAP nur teilweise verwirklicht (Abbildung 3.1). Insbesondere fehlen einige für die iterative Modelloptimierung wichtige Datenflüsse. Zwar stellt die realisierte Kopplung der Werkzeuge zu einer Werkzeugkette mittels Datei-Import und Datei-Export unter Zuhilfenahme von Transformatoren eine entscheidende Verbesserung gegenüber der traditionellen isolierten Werkzeugnutzung dar, jedoch können einige Möglichkeiten einer vollständigen Integration nicht genutzt werden. So ist zum Beispiel der Zugriff auf die mit dem CAD-System modellierten Grundrisse und das mittels des Layoutplanungssystem ermittelte Hallenlayout durch das Animationssystem derzeit nicht möglich. Geometrie und Layout müssen neu modelliert werden. Dieser Vorgang ist sehr zeitaufwendig und führt durch die Erzeugung redundanter Daten zu Konsistenzproblemen bei der Modifikation des Planungsmodells.

Ein entscheidendes Problem bei der Integration der auf Dateien basierenden Datenhaltungssysteme in einem FDBS ist das Fehlen eines Datenmodells mit festgelegter Semantik. Dadurch ist eine einfache, auf fundierten Methoden beruhende, Transformation in das kanonische Datenmodell nicht möglich. Ein vergleichsweise schwieriger Prozeß der Systemanalyse und nachträglichen Modellierung muß stattdessen durchgeführt werden. Bei der Untersuchung der durch den LOGAP genutzten Daten ist eine vereinfachte Modellierung entsprechend Abbildung 3.2 denkbar. Dabei wird auf die Darstellung der Einflußsphäre des Simulations- und Animationssystems verzichtet, da alle dargestellten Objekte für dieses System relevant sind. Gut erkennbar ist die intensionale Überlappung

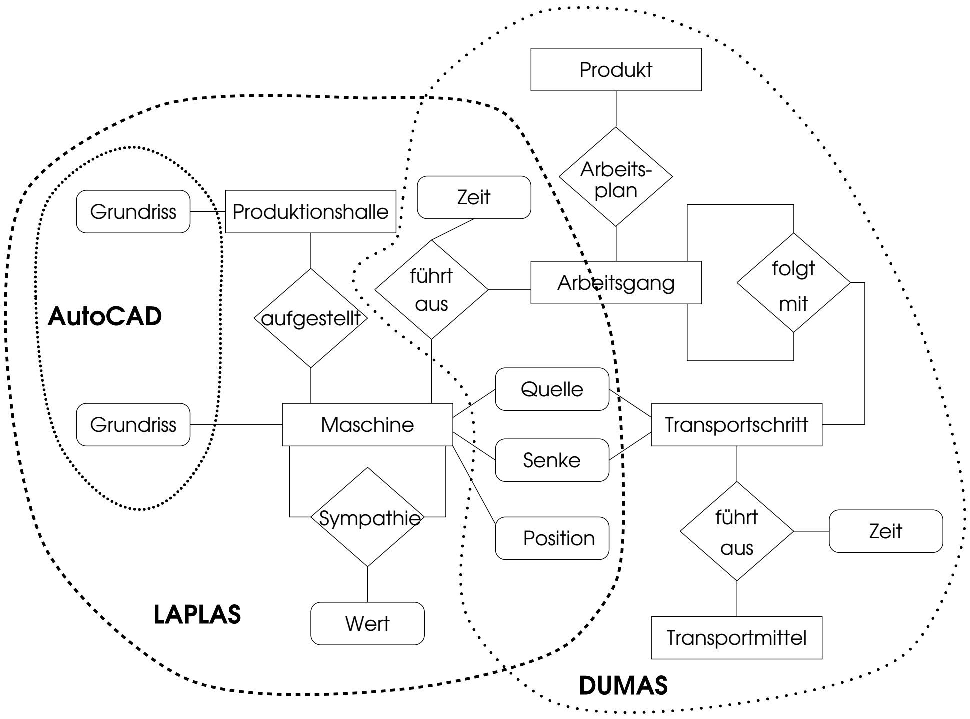

Abbildung 3.2: Beispielschema für den LOGAP

von Objekten und Eigenschaften der durch die verschiedenen Systeme benötigten Daten. Aus der Tatsache, daß der Planungsprozeß eine konkrete Fabrikanlage als Planungsgegenstand entwickelt, folgt weiterhin, daß eine Überlappung der Extensionen gegeben ist. Aus demselben Grunde ist zu vermuten, daß die Bestimmung der extensionalen Überlappung mittels Stringvergleich [Klu95] möglich ist.

Trotz der aufgezeigten Defizite bringt der Einsatz des LOGAP bei der Betriebsuntersuchung beachtlich Ergebnisse [FS94], die auf die Potentiale einer vollständigen Integration hinweisen. Zentrales Problem ist neben der technischen Realisierung des Zugriffs auf vom PC erzeugte Dateien vor allem die Modellierung der durch die Spezialapplikationen gehaltenen Daten. Mobilität ist eine weitere wichtige Anforderung für den Einsatz solcher integrierter Fabrikplanungstools, da viele Planungs- und Beratungsaufgaben, insbesondere die Modellierung, vor Ort mit dem Anlagenbetreiber zu lösen sind.

3.2.2 Beispiel 2: Simulatorentwicklungsumgebung Create!

Für die Optimierung komplexer logistischer Modelle sind analytische Methoden in vielen Fällen nur schwer anwendbar. Die geschlossene Darstellung des Modells wird mit zunehmender Komplexität schwieriger. Durch notwendige Vereinfachungen kommt es zu ungenauen und unsicheren Ergebnissen. Deshalb werden Abläufe in komplexen Systemen durch den Einsatz von Simulation untersucht. Dabei bilden Simulationsmodelle das dynamische Systemverhalten über fein nuancierte Randbedingungen und variable Systemlasteinstellungen detailliert ab. Sie erlauben Rückschlüsse auf zweckmäßige Steuerstrategien, Bestände sowie individuelle Auftrags- und Teiledurchlaufzeiten.

Das IFSL[2] ist an der Entwicklung der Simulatorentwicklungsumgebung Create! [Rüg95, RB95] beteiligt. Create! dient der Entwicklung bausteinorientierter Simulatoren, deren Schwerpunkt in der strategischen Gestaltung von Unternehmensprozessen sowie in der Simulation von Fabrikstrukturen mit Informationsflüssen und Steuerungsregeln liegt. Eine auf Basis von Create! entwickelte Anwendung zur Modellierung und statischen Analyse von Prozessen ist z.B. das Produkt Create!-Struktur. Auf Basis einer speziellen, mit Create! erzeugten Bausteinbibliothek können Simulationsmodelle für Fabrikstrukturen generiert werden. Die Durchführung von Simulationsexperimenten und die Animation der Experimentresultate wird dadurch möglich.

Die Datenhaltung von Create! und von Create!-Applikationen basiert auf Dateien, die hierarchisch im Dateisystem organisiert sind. Es werden u.a. Daten zur funktionalen und strukturellen Beschreibung von Bibliotheksbausteinen, zur Projekt-, Modell- und Experimentverwaltung gehalten. Eines der Entwicklungsziele ist die Umstellung von Create! bzw. Create!-Applikationen von einem dateibasierten Datenhaltungskonzept auf ein objektorientiertes Datenbanksystem. Dadurch wäre es möglich, die Funktionalitäten moderner Datenbankmanagementsysteme [Cod82] nutzen zu können. So ist es durch das Transaktionskonzept möglich, den parallelen konkurrierenden Zugriff auf die Daten zu nutzen. Die Anwendung mächtiger deskriptiver Anfragesprachen offeriert neue Methoden zur Auswertung von Simulationsexperimenten.

Bei einer solchen Umstellung ist es wünschenswert, die bereits vorhandenen Bausteinbibliotheken, Simulationsmodelle, etc. auch mit der neuen, auf einem Datenbanksystem basierenden Softwareversion nutzen zu können. Dabei sollte es nicht zur Unterbrechung langlaufender Simulationsexperimente kommen. Zur Erfüllung dieser Forderung kann ein föderiertes Datenbanksystem genutzt werden. Die Unterstützung der sanften Migration [BS95] von Daten und Applikationen durch FDBS wurde in [RS94, RS95] eingehend diskutiert. Durch ein solches Vorgehen werden die Risiken der Umstellung minimiert und somit die Erfolgsaussichten erhöht.

Der Einsatz des FDBS beginnt mit der Adaption des föderierten Schemas auf Basis der von Create! vorgegebenen Objektstruktur. Kommt im FDBS ein objektorientiertes kanonisches Datenmodell zum Einsatz, kann dieser Modellierungsschritt vergleichsweise einfach sein, da bei der Entwicklung von Create! ebenfalls objektorientierte Konzepte

[2] Institut für Förder- und Baumaschinentechnik, Stahlbau, Logistik der Otto-von-Guericke Universität Magdeburg

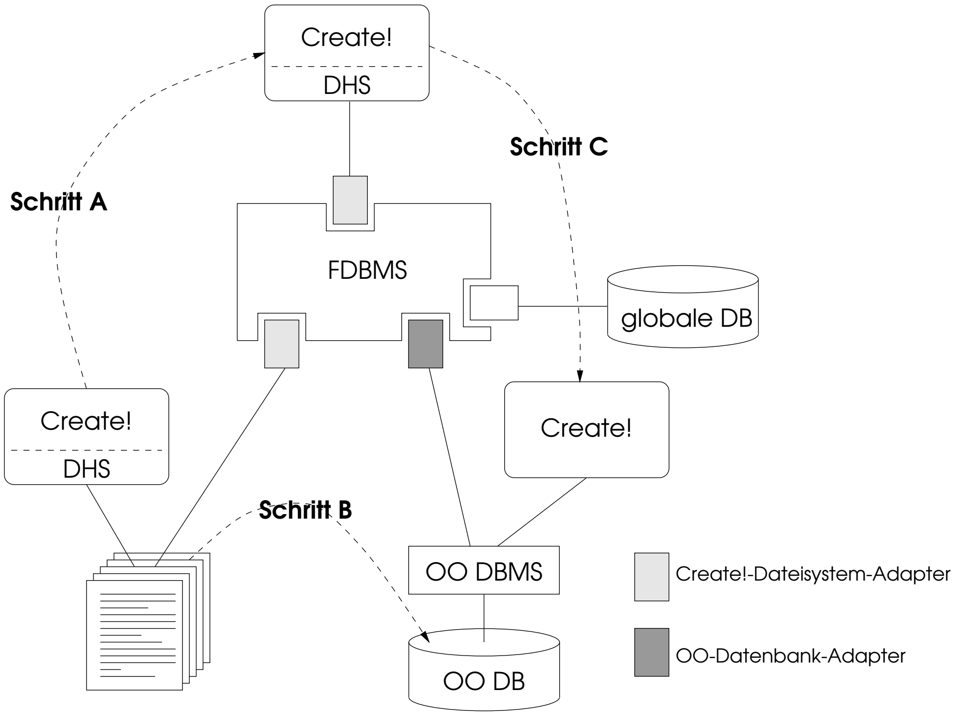

Abbildung 3.3: Migrationsszenario für Create!

Verwendung finden. Im Unterschied zum zuvor beschriebenen Beispiel ist eine Schemaintegration hier nicht notwendig, da nur ein Komponentendatenhaltungssystem vorhanden ist. Ein Datenbankadapter übernimmt die Abbildung der Daten aus dem Dateisystem auf das föderierte Schema. Als nächstes ist ein externer Adapter zu entwickeln, der eine inverse Daten- und Schemaabbildung vornimmt und somit auf der externen Ebene des FDBS den Zugriff äquivalent dem Zugriff auf das lokale Dateisystem realisiert. Damit ist die Migration der Create!-Applikation von der lokalen Schnittstelle auf die globale Schnittstelle möglich (Abbildung 3.3, Schritt A). Wenn das kanonische Datenmodell gleich dem Datenmodell des Zieldatenbanksystems ist (also wie in diesem Beispiel beide objektorientiert), so kann das föderierte Schema unmittelbar als lokales Schema dienen. An diesem lokalen Schema können dann Erweiterungen und Verbesserungen vorgenommen werden. Damit ist es möglich, die Daten aus dem Dateisystem sanft in das neue Datenbanksystem zu migrieren (Schritt B). Dies kann durch explizite Anweisungen in der Datenmanipulationssprache des kanonischen Datenmodells geschehen. Eine andere Möglichkeit besteht darin, grundsätzlich alle durch Create!-Applikationen über die globale Schnittstelle gelesenen oder geschriebenen Daten in der neuen Datenbank abzulegen. Weiterhin kann auf Basis des neuen Datenbanksystems eine Softwareversion entwickelt werden (Schritt C). Sind alle relevanten Daten migriert und arbeitet die neue Softwareversion stabil, kann in der weiteren Arbeit auf den Einsatz des alten Datenhaltungssystems, des Föderierungsdienstes und der alten Softwareversionen verzichtet werden.

Das beschriebene Vorgehen erlaubt die Nutzung der Datenbankfunktionalitäten bereits zu einem sehr frühen Zeitpunkt, und nicht erst nach Abschluß einer vollständigen

Migration. Weiterhin wird der integrative Einsatz zusätzlicher Applikationen auf Basis des FDBS bzw. des neuen lokalen DBS möglich, da die Integritätssicherung nun nicht mehr durch eine Create!-Applikation, sondern durch das DBMS realisiert wird und DBMS weitreichende Konzepte zur Unterstützung der Mehrbenutzerbetriebs bieten.

Auch in diesem Beispielszenario treffen wir wieder auf das Problem der Einbeziehung eines dateibasierten Datenhaltungssystems. Allerdings ist zu vermuten, daß die nachträgliche Modellierung aufgrund der vorhandenen Entwicklungsunterlagen und der genutzten objektorientierten Konzepte im Vergleich zum Beispiel 1 deutlich einfacher möglich ist.

3.3 Zusammenfassung

Bei der Darstellung der Projektziele und in den vorgestellten Beispielen sind einige spezifische Probleme und Anforderungen aufgeführt, die sich für die Anwendung föderierter Datenbanksysteme im Bereich der Fabrikplanung ergeben. Zentrale Bedeutung hat die integrierte Nutzung der Daten der vorhandenen Softwarewerkzeuge bei Wahrung der lokalen Autonomie und die konzeptionelle Verbesserung bestehender Systeme durch Unterstützung der sanften Migration von Daten und Applikationen. Zur Einbeziehung lokaler Datenhaltungssysteme, die auf Dateien basieren, ist eine nachträgliche Modellierung durchzuführen. Durch die lokalen Datenhaltungssysteme werden im allgemeinen keine Mechanismen zur Konsistenzsicherung und zur Unterstützung des Mehrbenutzerbetriebs angeboten. Diese Funktionalitäten können durch globale Mechanismen, die Teil des Föderierungsdienstes sind, nachgebildet werden.

Kapitel 4

Relevante Standards und Systeme

In diesem Kapitel werden wichtige Standards und Systeme vorgestellt, die für das Projekt relevant sind.

4.1 Standards

Die folgenden Abschnitte erläutern Standards, die bei der Föderierung heterogener Datenbanksysteme beachtet werden sollten. Dazu werden Ansätze zur Kommunikation (CORBA, RDA), zum einheitlichen Zugriff (ODBC, SQL, ODMG), zur Transaktionsverwaltung (X/Open - DTP) und zum Datenaustausch (STEP) im Bereich von Datenbank- und Datenhaltungssystemen vorgestellt.

4.1.1 OMA/CORBA

Die *Object Management Group* (OMG), ein Konsortium führender Soft– und Hardwarehersteller, hat mit der *Object Management Architecture (OMA)* eine Referenzarchitektur für verteilte objektorientierte Anwendungen in heterogenen Umgebungen vorgestellt (vgl. [Gro91]).

Die Grundlage dieser Architektur bildet das ebenfalls von der OMG spezifizierte *Core Object Model* [Gro95]. Dieses Modell definiert wesentliche Konzepte der Objektorientierung, wie Objekt, Operation, Typ oder Subtyping. Jedes OMG–konforme Produkt muß mindestens dieses Core Model implementieren. Darüber hinaus können Erweiterungen definiert werden, die in sogenannten *Profiles* zusammengefaßt werden. Solche Erweiterungen sind z.B. das ODBMS–Profile der ODMG mit Konzepten wie Persistenz, Relationships, Transaktionen und das ORB–Profile mit dem Konzept des entfernten Operationsaufrufs.

In Abbildung 4.1 sind die OMA–Komponenten dargestellt. Ein *Object Request Broker (ORB)* ist die zentrale Komponente dieser Architektur. Der ORB ermöglicht netzwerktransparente Aufrufe von Operationen eines Objektes über Prozeß–, Programmiersprachen– und Betriebssystemgrenzen hinweg. Er sorgt für Transparenz

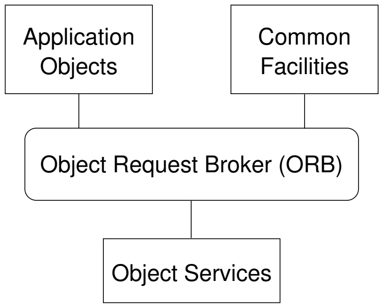

Abbildung 4.1: Object Management Architecture [Gro91]

bezüglich der Lokation der Serverobjekte und stellt Dienste zur Registrierung, zum Auffinden und zur Aktivierung von Servern bereit.

Die *Object Services* stellen eine Sammlung von Basisdiensten dar. Von der OMG wurden im Rahmen der *Common Object Service Specification (COSS1)* [Gro94] u.a. Dienste zur persistenten Speicherung von Objektdaten, zur Namensverwaltung, zur Definition und Verwaltung von Beziehungen zwischen Objekten und zur asynchronen Ereignismeldung spezifiziert.

Die *Common Facilities* sind Objekte, die allgemein nutzbare, höherwertige Dienste bereitstellen. Sie lassen sich in zwei Kategorien unterteilen: die *Vertical Market Facilities* für spezielle Marktsegmente und die *Horizontal Common Facilities* für die Anwendung in mehreren Problembereichen.

Schließlich entsprechen die *Application Objects* den klassischen Anwendungen, die Dienste anderer Objekte in Anspruch nehmen.

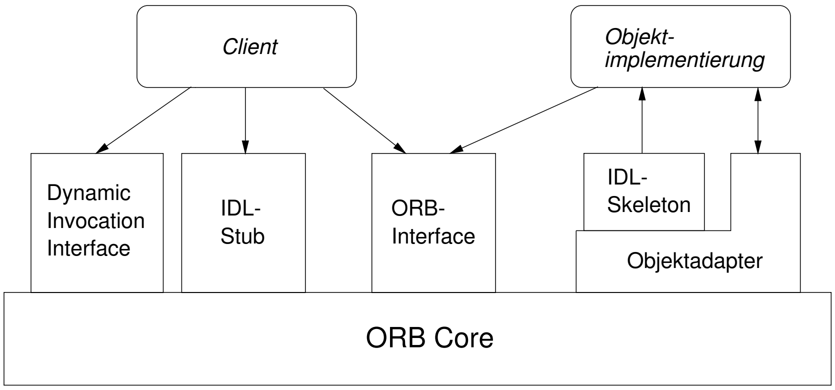

Abbildung 4.2: Common Object Request Broker Architecture [Gro91]

Mit der *Common Object Request Broker Architecture (CORBA)* hat die OMG eine Spezifikation eines konkreten ORB vorgestellt (Abbildung 4.2). Hierbei wird das bekannte Client–Server–Konzept auf Objektebene realisiert. Ein Serverobjekt (eine sogenannte *Objektimplementierung*) kann einem Client Dienste zur Verfügung stellen. Der Zugriff auf diese Dienste ist nur über eine wohldefinierte Schnittstelle möglich. Zur Sicherstellung der Programmiersprachenunabhängigkeit von Client und Objektimplementierung wird diese

Schnittstelle in der *Interface Definition Language (IDL)* spezifiziert. IDL ist an C++ angelehnt, hat jedoch ausschließlich beschreibenden Charakter und enthält keine Anweisungskonstrukte. Aus der IDL–Spezifikation können sogenannte Stubs für die Client–Seite bzw. Skeletons für die Server-Seite generiert werden. Die Abbildung von IDL in eine Implementierungssprache ist bisher für C, C++ und Smalltalk definiert.

Insgesamt stellt OMA/CORBA einen geeigneten Ansatz eines herstellerübergreifenden Standards für verteilte objektorientierte Anwendungen in heterogenen Umgebungen dar. Erste Produkte sind u.a. von IBM (DSOM), Sun (NEO), Iona (Orbix) und HP (Distributed Smalltalk) verfügbar. Ein Problem ist gegenwärtig noch die Interoperabilität zwischen ORBs verschiedener Hersteller. Die neue Version 2.0 des CORBA–Standards soll hier durch die Definition eines Inter–ORB–Protokolls Abhilfe schaffen.

4.1.2 RDA

RDA (*Remote Database Access*) [Sun93, Rah94] ist einer von internationalen Standards, die die Kommunikation zwischen Computersystemen beschreiben. Er ist in der Applikationsschicht des ISO/OSI-Referenzmodells angesiedelt. Ziel ist die einfache Kopplung von Applikationen und Datenbanksystemen unterschiedlicher *Hersteller, Art bzw. Komplexität.*

Da eine Applikation selbst ein Datenbanksystem sein kann, kann RDA zur Kommunikation unterschiedlicher Datenbanksysteme untereinander verwendet werden.

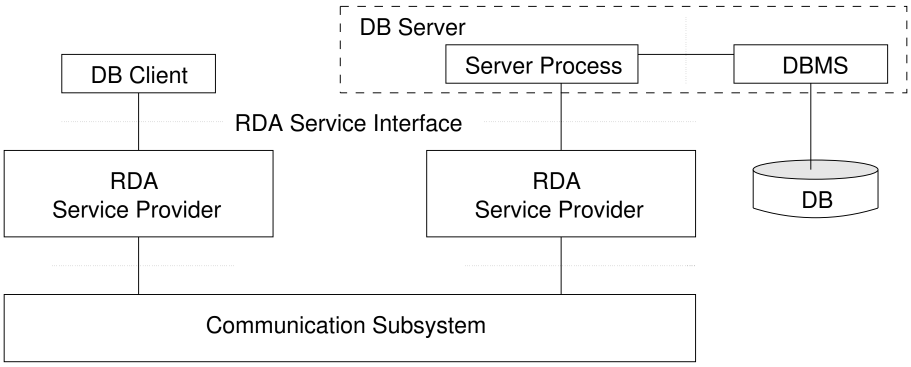

Abbildung 4.3: RDA Client/Server Model

Im Modell in Abbildung 4.3 steht der **RDA Service** dem **DB Client** und dem **DB Server** zur Verfügung. Der RDA Service nutzt dabei die Dienste der durch ISO/OSI spezifizierten Subschichten (Schichten 1 bis 6 des ISO/OSI-Referenzmodells) zur Kommunikation (*Communication Subsystem*). Auf der DB Server-Seite läuft ein **Server Process**, der die ankommenden Anforderungen des DB Clients durch das DBMS bearbeitet und Ergebnisse über den RDA Service zurückliefert.

Die Dienste des RDA Service werden nach ihrer Funktionalität in fünf verschiedene Gruppen eingeteilt:

1. *Dialogue Management Services* (z. B. `R_BeginDialog, R_EndDialog`)

2. *Resource Handling Services*, zur Zuweisung und Lokalisierung von Ressourcen, wie z. B. Datenbank, Tabelle, Tupel (z. B. `R_Open, R_Close`)
3. *Database Language Services* (DBL Services)
4. *Control Services*, Abfragen und Änderungen des Status der Datenbankoperationen des DB-Servers vom DB-Client aus (z. B. `R_Cancel_Request, R_Close_Request`)
5. *Transaction Management Services*, (z. B. `R_Begin_Transaction, R_Commit`)

Diese Funktionen, die dem DB-Client und DB-Server zur Verfügung stehen, werden im **RDA Service Interface** definiert. Seit der Annahme von RDA durch die ISO im Jahre 1986 (SC21 WG3, Database) haben sich zwei Teile innerhalb der ISO/OSI-Applikationsschicht entwickelt. Der erste Teil ist *RDA Generic*, in dem datenbankunabhängige Aspekte der RDA Client/Server– Kommunikation beschrieben werden. Im zweiten Teil, der *RDA Specialization* werden die datenbankspezifischen Details, wie z. B. *SQL–Specialization*, beschrieben.

Das RDA Client/Server–Modell kann im Bereich von FDBS für die Kommunikation mit CDBS verwendet werden. Dabei kann auf Seiten des DB Servers eine Offenlegung von lokalen Transaktionen erfolgen, um diese in globale Transaktionen einzubeziehen.

4.1.3 ODBC

ODBC (Open Database Connectivity) ist Microsoft's Strategie der Bereitstellung einer einheitlichen Zugriffsschnittstelle auf heterogene relationale DBMS oder Dateisysteme unter DOS/Windows [Lam94]. Zur Realisierung dieses einheitlichen Zugriffs wird für jeden Typ einer Datenbasis ein Treiber (ODBC Treiber) angeboten. Die ODBC Architektur wird in Abbildung 4.4 deutlich.

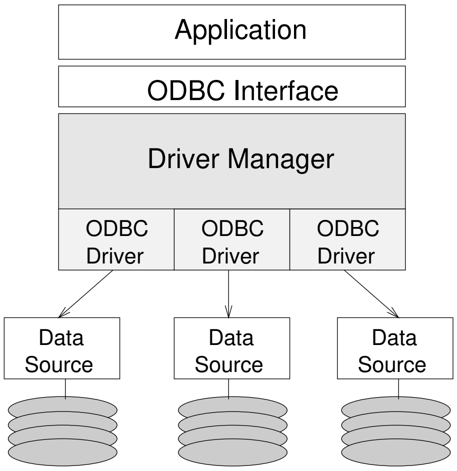

Abbildung 4.4: ODBC Architektur nach [Lam94]

In der **Application** erfolgen Aufrufe von ODBC Funktionen mit Übergabe- und Rückgabeparametern. Diese Funktionen werden vom **Driver Manager** durch das **ODBC**

Interface zur Verfügung gestellt. Der Driver Manager hat weiterhin die Aufgabe, die einzelnen **ODBC Driver** zu verwalten. Die für jede spezielle Art einer Datenbasis (**Data Source**) existierenden Treiber arbeiten die aufgerufenen ODBC Funktionen ab und stellen die Ergebnisse bereit. Der SQL-basierte Zugriff auf Daten erfolgt dabei durch die Einbettung von SQL-Anfragen als Parameter solcher ODBC Funktionen. Die ODBC Treiber der einzelnen Systeme werden auf zwei Arten unterschieden. Die eine Art ist die *ODBC Funktionalität*, die die Art und Vielfalt der in der ODBC Schnittstelle angebotenen Funktionen angibt. Diese Funktionalität wird in *Core, Level 1* und *Level 2* unterteilt. Die zweite Art ist die Unterscheidung nach der *SQL Funktionalität* in *Minimum, Core* und *Enhanced*. ODBC Treiber stehen beispielsweise für MS-Access, Paradox, Btrieve, dBase (Windows Datenbanken) und Objectivity/DB 3.0 WIN, Oracle Rdb-V2.0, Sybase Open Client, DB/2 (Windows Clients) zur Verfügung.

Durch die Bereitstellung von ODBC Treibern (derzeit ODBC 2.0) für verschiedene Datenhaltungssysteme unter Windows wird ein einheitlicher Zugriff gewährleistet, der bei der Einbeziehung von Windows-DBS innerhalb von FDBS interessant erscheint. Ebenso existieren Applikationen, die ODBC als Zugriff auf Daten verwenden. ODBC ist demnach ein Kandidat für eine externe Schnittstelle.

4.1.4 SQL

Relationale Datenbanksysteme sind derzeit am weitesten verbreitet und stellen einen Standard für die Verwaltung großer Datenmengen dar (z. B. Oracle, DB2, Ingres, Informix, Sybase). Die Datenbanksprache dieser relationalen Datenbanken ist SQL (Structured Query Language), die 1986 von der ANSI genormt (SQL86) [Dat87], 1989 von der ISO übernommen und zu SQL89 ergänzt wurde und seit 1992 als SQL2 vorliegt [DD93]. Derzeit ist ein SQL3 [Pis93] in Vorbereitung. SQL definiert eine Datendefinitionssprache (DDL) sowie eine Datenmanipulationssprache (DML) zur Beschreibung der Daten, Angabe von Integritätsbedingungen, zum Einfügen, Löschen, Ändern und weiterhin zur Absetzung deklarativer Anfragen, zur Definition von Sichten, Verwaltung von Indizes, Vergabe von Rechten etc. Weiterhin realisieren einige Anbieter relationaler Datenbanksysteme ein Triggerkonzept zur Wahrung von Integrität bei bestimmten Ereignissen (Oracle 7, Sybase). Neben interaktiven SQL–Schnittstellen der jeweiligen Datenbankserver kann SQL zur Entwicklung von Applikationen auch in Programmiersprachen eingebettet werden, wie z. B. C oder PASCAL.

Wegen der großen Verbreitung von SQL-Datenbanken mit standardisierter Datenbanksprache SQL stellen diese Datenbanken wichtige Kandidaten für die Einbindung in ein föderiertes System dar. Die dabei realisierten Triggerkonzepte spielen eine wichtige Rolle in Bezug auf die aktive Konsistenzsicherung im Projekt **SIGMA**$_{FDB}$.

4.1.5 ODMG-93

Die ODMG (*Object Database Management Group*) ist eine Untergruppe der OMG und wurde 1991 von führenden Herstellern von ODBMS mit dem Ziel der Standardisierung von objektorientierten Datenbanksystemen gegründet. Der derzeit veröffentlichte Standard ODMG-93 (Version 1.0) [Cat94] definiert:

1. **Objektmodell**

   Das Objektmodell beinhaltet eine Reihe von gängigen objektorientierten Modellierungskonzepten wie *Objekt, Objekttyp, Objektidentität, Attribute, Beziehungen, Vererbung* sowie datenbankspezifische Erweiterungen wie *Schlüssel* und *Extentionen.*

2. **Objekt-Beschreibungssprache (ODL)**

   Die ODL (Object Definition Language) stellt eine programmiersprachenunabhägige Beschreibungsmöglichkeit des Objektmodells zur Definition von Datenbank-Schemata zur Verfügung.

3. **Objekt-Anfragesprache (OQL)**

   Die OQL (Object Query Language) ist eine an das SQL relationaler Systeme angelehnte Anfragesprache über dem Objektmodell. Es gibt sowohl Erweiterungen als auch Einschränkungen gegenüber SQL. Die OQL ist eine mächtige Anfragesprache und erlaubt einen einfachen, SQL-ähnlichen, deklarativen Zugriff auf Objektdatenbanken.

4. **Sprachanbindungen**

   Der ODMG-93 Standard beschreibt eine C++- sowie eine SmallTalk-Sprachanbindung.

Eine ODMG-93-Schnittstelle wird derzeit nur von wenigen ODBMS-Herstellern angeboten. Auch weist der aktuelle Standard einige Fehler, Inkonsistenzen und Mängel auf, die in der nächsten Version 1.1 behoben werden sollen. Der ODMG-Standard dient also zur Vereinheitlichung der Modellierung und des Zugriffs auf unterschiedliche Objektdatenbanken, die Kandidaten für die Einbeziehung in FDBS sind. Dabei bietet sich das semantisch reiche Objektmodell als *kanonisches Datenmodell* des FDBS an.

4.1.6 X/Open - DTP

Das X/Open - DTP (*Distributed Transaction Processing*) Modell beinhaltet eine Beschreibung zur Offenlegung von Transaktionen von Datenhaltungssystemen [Spe90, Spe92]. Dabei wird zwischen drei Software-Komponenten unterschieden:

- Das *Application Programm (AP)*, in dem die Transaktionsklammern und Aktionen der Transaktion definiert werden.

- Die *Resource Manager* (*RMs*, z. B. Datenbanken oder Dateisysteme) für den Zugriff auf gemeinsam verwendete Ressourcen.

- Ein *Transaction Manager (TM)* zur Verwaltung globaler Identifikatoren für Transaktionen, zur Statusüberwachung, zur Realisierung von Commit-Protokollen und zur Fehlerbeseitigung.

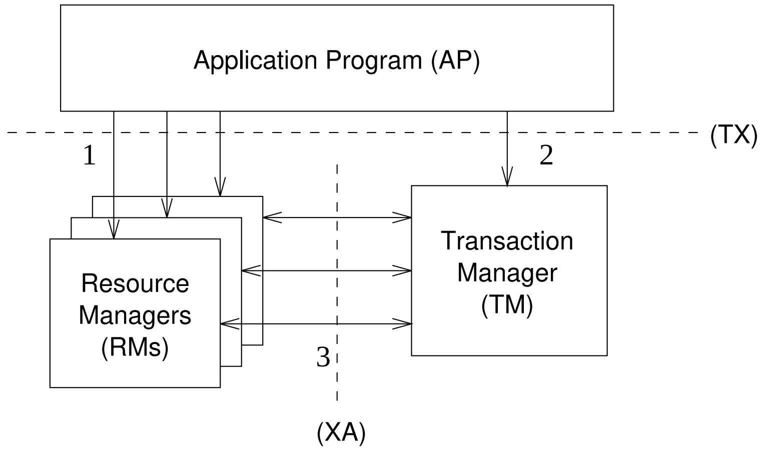

Abbildung 4.5: X/Open DTP Model

Abbildung 4.5 stellt die Komponenten des DTP Systems dar, die untereinander kommunizieren. Die drei Kommunikationsschnittstellen beinhalten folgendes:

1. Durch die AP–RM–Schnittstelle kann die Applikation *lokale Transaktionen* des RM beeinflussen (initialisieren, starten, beenden), wie es aus der Applikationsprogrammierung lokaler Datenbankanwendungen bekannt ist.

2. Zur Unterstützung *globaler Transaktionen* wird die AP-TM-Schnittstelle spezifiziert. Die Applikation (AP) verwendet Routinen des TM zum Starten, Beenden, etc. von globalen Transaktionen.

   Beide Schnittstellen (AP-RM und AP-TM) werden in [Spe92] spezifiziert und als TX-Schnittstelle bezeichnet.

3. Die XA-Schnittstelle [Spe90] spezifiziert die Schnittstellen zwischen RM und TM. Der TM benachrichtigt die RM zur Einordnung ihrer lokalen in die globalen Transaktionen, realisiert das globale 2-Phasen-Commit-Protokoll sowie Transaktionsrücksetzungen. Dabei sind die RM einem TM zugeordnet. Für einen RM wird spezifiziert, welche Routinen dem TM zur Verfügung gestellt werden.

Das X/Open–DTP spezifiziert eine offene Transaktionsverwaltung von globalen Transaktionen unter Einbeziehung lokaler Transaktionen verschiedener Datenhaltungskomponenten. Die TX-Schnittstelle ist demnach ein Kandidat für die Integration von Transaktionen im Rahmen der Föderierung heterogener Datenhaltungskomponenten. Die XA-Schnittstelle kann intern zur Kopplung der globalen mit den lokalen Transaktionen verwendet werden.

4.1.7 STEP

Der ISO–Standard 10303 [TC193, GAP93], besser bekannt unter dem Namen STEP – *Standard for the Exchange of Product Model Data*, ist ein Ansatz zur Normung eines integrierten Produktmodells, das Produktinformationen aus allen Phasen des Produktlebenszyklus abbildet. STEP soll damit die Probleme traditioneller Schnittstellen zur Übertragung von Produktdaten (wie Redundanz, Unvollständigkeit usw.) beheben. Der Standard beinhaltet im einzelnen:

Beschreibungsmethoden: Zur Spezifikation von Informationsmodellen wurde die formale Beschreibungssprache EXPRESS und die graphische Notation EXPRESS–G definiert.

Implementierungsmethoden: Zur eindeutigen Abbildung der spezifizierten Informationsmodelle auf verschiedene Implementierungsformen sind folgende Methoden festgelegt:

- Austausch von Produktdaten über Dateien
- Verwendung einer Programmierschnittstelle zum Zugriff auf Modelldaten
- Daten–Sharing über STEP–Datenbanken

Test– und Prüfmethoden: Zur Prüfung der Konformität der Implementierungen gegenüber der Spezifikation sind spezielle Methoden festgelegt.

Allgemeine Basismodelle: Dies sind Informationsmodelle, die Produktdaten unabhängig von bestimmten Anwendungen spezifizieren.

Anwendungsspezifische Basismodelle: Dies sind anwendungsspezifische Erweiterungen der allgemeinen Basismodelle.

Applikationsprotokolle: Ein Anwendungsprotokoll beschreibt die Verwendung der Basismodelle in einem bestimmten Anwendungskontext.

Mit STEP steht somit eine internationale Norm zur Verfügung, die die Bereitstellung und Verarbeitung von Produktdaten unterstützt und Methoden zur Entwicklung integrierter Produktmodelle definiert. Für die Entwicklung STEP–basierter Applikationen stehen eine Reihe von Werkzeugen zur Verfügung, die durch die Bereitstellung einer externen STEP–Schnittstelle im Projekt $\mathbf{SIGMA}_{FDB}$ genutzt werden können.

4.2 Systeme

Die folgenden Abschnitte stellen Systeme im Bereich föderierter Datenbanken vor, die bereits verfügbar sind (Omni SQL), als Prototyp vorliegen (OpenDM/Efendi, IRO-DB, ZOO_{IFI} oder Ansätze darstellen (FIHD).

4.2.1 Omni-SQL-Server

Der Omni-SQL-Server ist ein Produkt von Sybase. Er ist ein eigenständiger SQL-Server mit Merkmalen wie Nutzerverwaltung, lokaler Datenhaltung, Transaktions- und Integritätsmechanismen. Neben diesen Eigenschaften ermöglicht der Omni-SQL-Server den Zugriff auf Daten anderer SQL-Server wie *Sybase-, Oracle-, DB2- und Ingres SQL-Server*.

Die Integration verschiedener SQL-Server wird in Abbildung 4.6 verdeutlicht.

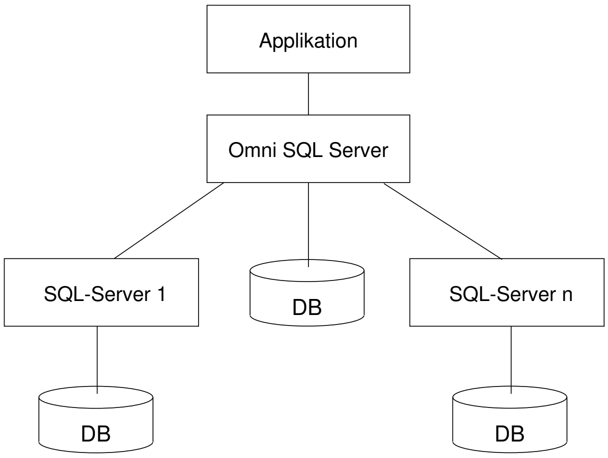

Abbildung 4.6: Integration durch Omni-SQL-Server

Im Forschungsgebiet föderierter Datenbanksysteme ist besonders der einheitliche Zugriff auf andere SQL-Server interessant. Durch die Kopplung können Tabellen beliebiger Lokalität vom Omni-SQL-Server aus verwendet werden, wobei die Autonomie der beteiligten SQL-Server gewahrt wird.

4.2.2 OpenDM/Efendi

OpenDM/Efendi wird von Cadlab (Kooperation der Universität Paderborn mit der Siemens-Nixdorf Informationssysteme AG) mit dem Ziel der Kopplung verschiedener heterogener Datenbank- und Dateisysteme zu einem föderierten Datenbanksystem entwickelt [Rad94]. Der derzeitige Prototyp realisiert die Kopplung eines objektorientierten DBS, eines relationalen DBS und des UNIX-Dateisystems mit spezifischem Datei-Formaten. Dabei wird ein einheitlicher transparenter Datenzugriff unter Wahrung der Autonomie und Konsistenz gewährleistet. Die Schemaintegration erfolgt nach der Vorgehensweise von [SL90]. Zur Erstellung globaler Applikationen wird eine ODMG-93-Schnittstelle [Cat94] mit erweiterter datenbankspezifischer Funktionalität und eine Schema-Integrationssprache angeboten. Neben diesen Konzepten wird die Funktionalität der Datenmigration zwischen verschiedenen DBS zur Verfügung gestellt.
Die Architektur des FDBS Efendi wird in Abbildung 4.7 verdeutlicht.

Ein Datenbankadapter (*DBS Adapter*) realisiert die Kommunikation mit einem spezifischen Datenbank- bzw. Dateisystem. Durch die Adapter erfolgt die Transformation

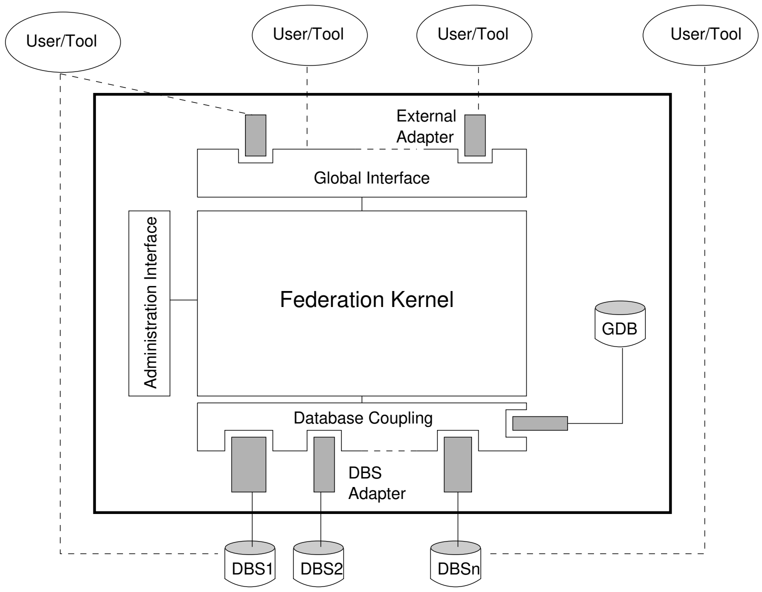

Abbildung 4.7: Architektur des FDBS Efendi

der Daten und Operationen des CDBS in das kanonische Datenmodell von Efendi. Durch die Kapselung der Datenbankspezifika in diesen Adaptern ist der Kern des Systems unabhängig von der Art des verwendeten DBS. Zur dynamischen Kopplung der Adapter wird die *Database Coupling*-Schicht verwendet. Der *Federation Kernel* ist eine Datenbankmaschine mit den typischen Datenbankkomponenten wie Data Dictionary, Cache, Anfragebearbeitung, globalem Transaktionsmanagement sowie globaler Integritätskontrolle. Meta-Informationen werden in einer globalen Datenbank (*GDB*) verwaltet. Zur Administration des FDBS, zu der das Starten, Stoppen, Intitialisieren und Koppeln der Komponenten-Datenbanken gehört, steht ein *Administration Interface* zur Verfügung. Globale Datenzugriffe erfolgen über das *Global Interface*, welches eine Erweiterung der ODMG–C++–Schnittstelle um Schemaintegration und Datenmigration darstellt. Geplant sind *External Adapters*, die z. B. applikationsspezifische Schnittstellen (SQL, SDAI, etc.) oder Sichten anbieten.

OpenDM/Efendi realisiert einen im Rahmen des Projekts $\mathbf{SIGMA}_{FDB}$ interessanten Prototypen. Durch eine geplante Zusammenarbeit mit Cadlab können wesentlicher Erfahrungen auf diesem Gebiet ausgetauscht werden.

4.2.3 $\mathbf{ZOO}_{IFI}$

ZOO_{IFI} (***Z**urich **O**bject-**O**riented **I**ntegration **F**ramework for Bu**i**lding Heterogeneous Database Systems*) ist ein Projekt des Institutes für Informatik an der Universität Zürich [HD93, Där94]. Ziel dieses Projektes ist die Entwicklung eines Basissystems (ZOO_{IFI}) zur flexiblen Integration eines breiten Spektrums von Datenhaltungssystemen in einer Föderation. Resultat dieser Integration ist ein Homogenisiertes Datenbanksystem (HDBS) mit einem globalen objektorientierten Datenmodell. Die objektorientierte Schnittstelle bietet neben den herkömmlichen Konzepten:

- spezielle Unterstützung komplexer Objekte im Kontext der Föderation
- verschiedene Integrationsalternativen zur Repräsentation lokaler Daten
- globale Objektidentität und
- Konzepte zur Spezifikation verschiedener Klassen zur flexiblen und korrekten Abbildung auf die lokalen Datenhaltungssysteme

Hierzu existiert ein Prototyp, der eine solche Schnittstelle über die autonomen Datenhaltungssysteme realisiert. Es werden sowohl objektorientierte DBMS als auch Dateisysteme in die Föderation einbezogen. Die Methodiken und Werkzeuge des Züricher Institutes auf diesem Gebiet erscheinen interessant, zumal eine Vorgehensweise in Form eines 'Kochbuches' und verschiedene Werkzeuge bei der Föderierung von Datenhaltungssystemen vorgestellt werden.

4.2.4 IRO-DB

IRO-DB (*Interoperable Relational and Object Databases*) ist ein ESPRIT-III-Projekt der GMD-IPSI (*Institute for Integrated Publication and Information Systems*) Darmstadt in Zusammenarbeit mit GOPAS, IBERMATICA, EDS, GRAPHAEL, INTRASOFT, O_2 Technology und PRiSM [BFHK94, BFN94, GGF$^+$95]. Die Ziele dieses Projektes sind:

- Unterstützung eines objektorientierten Datenmodells unter Einbeziehung relationaler und objektorientierter Schemata lokaler DBMS
- Bereitstellung einer objektorientierten Anfragesprache
- transparenter Zugriff auf verteilte Daten
- Werkzeuge zur Unterstützung des Integrationsprozesses
- eine standardisierte Zugriffsschnittstelle (API)

IRO-DB ist ein Prototyp eines föderierten Datenbankmanagementsystems mit Client/Server–Architektur. Es föderiert relationale und objektorientierte Datenbanksysteme auf Basis des ODMG-Objektmodells. Dabei lassen sich folgende drei Schichten unterteilen (Abbildung 4.8):

- Lokale Schicht (*Local Layer*)
  Hier erfolgen die Abbildungen der lokalen Datenbank-Schemata in das gemeinsame Datenmodell und die Zugriffe auf die gespeicherten Daten durch die lokalen Datenbank-Adapter (LDA).

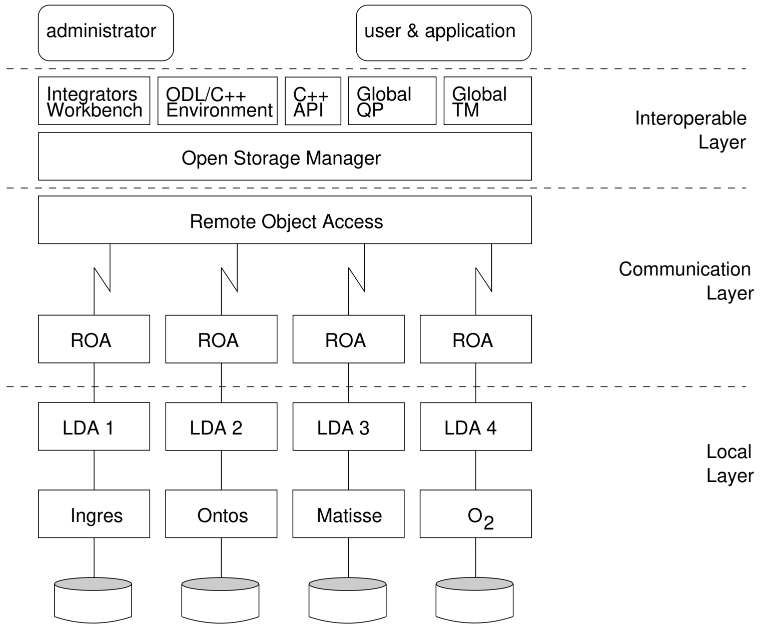

Abbildung 4.8: IRO-DBArchitektur nach [BuFaHu94]

- Kommunikationsschicht (*Communication Layer*)
  Die Kommunikationsschicht ermöglicht den transparenten verteilten Zugriff auf die gekoppelten Datenbanken.

- Operationsschicht (*Interoperable Layer*)
  Die Operationsschicht bietet einen transparenten integrierten Zugriff auf lokale Daten durch eine C++ API mit Anfragesprache (global QP) und globalem Transaktionsmanagement(global TM) sowie Administrations-Werkzeuge zur Unterstützung der Integration (Integrators Workbench mit C++ ODL).

IRO-DB befindet sich derzeit in der Entwicklung mit der Erreichung des Projektzieles 1996. Im Rahmen des Projektes ist interessant, wie konkrete kommerzielle Datenbanksysteme wie ONTOS, O_2, MATISSE und INGRES föderiert werden, wie die prinzipiell unterschiedliche Datenmodelle homogenisiert werden und in wieweit Werkzeuge diesen Integrationsprozeß unterstützen.

4.2.5 FIHD

Innerhalb der Zentralabteilung Forschung und Entwicklung der Siemens AG wird im Projektes FIHD - Flexible Integration Heterogener Datenbanksysteme [Hoh95, HK95, Wei95, Jan95] ein flexibler Integrationsrahmen für die Kopplung heterogener Datenbanksysteme entwickelt. Ziel des Projektes ist der integrierte Zugriff auf bereits bestehende Datenbestände, die von unterschiedlichen lokalen Datenbanksystemen verwaltet werden. Dabei soll die Autonomie der lokalen Datenbanksysteme und der darauf basierenden Anwendungen möglichst wenig beeinträchtigt werden. Abbildung 4 veranschaulicht die Architektur des FIHD-Ansatzes.

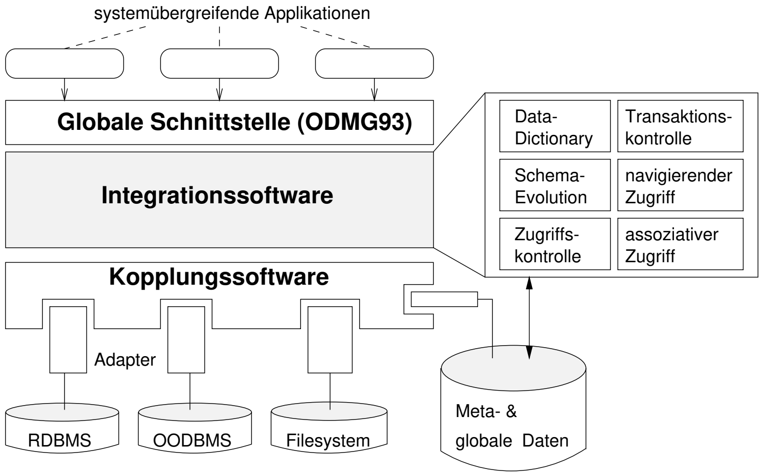

Abbildung 4.9: Architektur des FIHD Ansatzes

- Die *Kopplungssoftware* bildet den Rahmen zur Kopplung unterschiedlicher Datenhaltungssysteme. Durch entsprechende Adapter lassen sich Datenhaltungssysteme beliebig ein- und ausklinken. Die Aufgabe der Kopplungsschicht besteht im wesentlichen in der Beseitigung der Heterogenität bezüglich Datenmodellierung und Zugriff. Neben den lokalen Datenhaltungssystemen wird des weiteren eine Datenbank (in einem der lokalen Systeme oder in einem zusätzlichen System) zur Aufnahme von Metadaten und systemübergreifenden, globalen Daten benötigt.

- Die Integrationssoftware übernimmt die logische Integration der lokalen Datenbestände und stellt den systemübergreifenden Applikationen eine einheitliche Schnittstelle zur Verfügung. Entsprechende Datenbankfunktionalität wird, bezogen auf die Integration, bereitgestellt, so z. B. eine Transaktionssteuerung, die nun datenbankübergreifend realisiert wird oder ein assoziativer Zugriff, der mehrere Datenbanken unterschiedlicher Systeme einbezieht. Die Ortstransparenz wird gewährleistet, so daß eine Applikation vom physikalischen Aufbewahrungsort (Rechner, Datenhaltungssystem, Datenbasis, etc.) der Daten abstrahieren kann.

- Die globale Schnittstelle ermöglicht (neuen) Applikationen, sowohl auf einzelne Datenbasen als auch auf den integrierten Datenbestand in einheitlicher Form zuzugreifen. Die Schnittstelle basiert auf dem ODMG-93-Standard, dem angestrebten Standard für objektorientierte Datenbanksysteme.

FIHD stellt wie **SIGMA**$_{FDB}$ einen eng gekoppelten Ansatz zur Föderation heterogener Datenhaltungssysteme dar und ist auf dem Weg zu einer prototypischen Realisierung. Die dabei gewonnenen Erkenntnisse und Lösungeswege erscheinen auch im Projekt **SIGMA**$_{FDB}$ interessant.

Kapitel 5

Föderierter Datenbankentwurf

Ein wichtiges Anliegen des Forschungsprojektes $\mathbf{SIGMA}_{FDB}$ ist die Untersuchung von Methoden zur flexiblen Schematransformation und -integration. Ein zentrales Problem dabei ist die Wahl des kanonischen Datenmodells. Weiterhin betrachten wir die Einbeziehung von auf Dateien basierenden Datenhaltungssystemen in FDBS, den Einsatz gewichteter Integrationsstrategien und eine Klassifikation für Schemakonflikte.

5.1 Das kanonische Datenmodell

Betrachtet man den Entwurfsprozeß föderierter Datenbanksysteme, so wird die zentrale Bedeutung des kanonischen Datemodells offensichtlich. Das kanonische Datenmodell ist Zielmodell der modellhomogenisierenden Schematransformation, Basismodell der Schemaintegration und Ausgangsmodell der Heterogenisierung. In [BLN86] werden die Kriterien *Korrektheit, Vollständigkeit* und *Verständlichkeit* zur Bewertung föderierter Schemata formuliert. Anhand dieser Kriterien lassen sich mögliche kanonische Datenmodelle auf ihre Eignung untersuchen.

Gegeben seien verschiedene heterogene Datenbank- bzw. Datenhaltungssysteme, die sogenannten Komponentendatenbanksysteme mit ihren lokalen Schemata. Die CDBS basieren auf unterschiedlichen Datenmodellen, die in der Regel eine unterschiedliche Menge an Konzepten zur Modellierung zu Verfügung stellen.

Ein Beispiel für ein semantisch armes Datenmodell ist das relationale Modell, das nur über eine geringe Menge an Konzepten zur Modellierung verfügt. Aufgrund des theoretisch fundierten Modells und stabil arbeitender relationaler Datenbankmanagementsysteme ist es nach wie vor das vorherrschende Datenmodell für viele Standarddatenbankanwendungen.

Zu den Datenmodellen, die über einen großen Reichtum an Modellierungskonzepten verfügen, zählen die objektorientierten Datenmodelle. Durch Konzepte wie Klassen, Generalisierung, Spezialisierung, Beziehungen, komplexe Attribute und Objektidentität lassen sich komplexe Weltausschnitte wirklichkeitsnah und verständlich modellieren. Auf einem objektorientierten Datenmodell basierende Datenbanksysteme werden vor allem für Nichtstandardanwendungen, z.B. für CAD-Systeme, eingesetzt. Nachteilig ist das Fehlen eines standardisierten und theoretisch fundierten objektorientierten Datenmodells, das die Forderungen von [Cod82] und [ABD$^+$89] vollständig erfüllt.

Zur Überwindung der Datenmodellheterogenität werden die lokalen Schemata in das kanonische Datenmodell transformiert. In [SCG91, Kac95] wird die Verwendung eines semantisch reichen Datenmodells als kanonisches Datenmodell vorgeschlagen. Damit ist es möglich, lokale Schemata, die die weitreichenden Modellierungskonzepte semantisch reicher Datenmodelle nutzen, weitgehend korrekt und vollständig zu integrieren. Lokale Schemata in einem semantisch armen Datenmodell hingegen müssen bei der Transformation semantisch angereichert werden [Cas93, HK95]. Allerdings ermöglicht der Einsatz semantisch reicher Datenmodelle die Modellierung des gleichen Weltausschnittes auf vielfältige Art und Weise. Dadurch können viele Schemakonflikte auftreten, die während des Prozesses der Integration zu lösen sind. Die Integration erfordert die Umstrukturierung und Normalisierung der transformierten Schemata. Beispiele zeigen [GSCS95], daß korrekte und vollständige föderierte Schemata nicht zwangsläufig auch das Kriterium der Verständlichkeit erfüllen. Eine Ursache hierfür liegt in der, durch Verwendung semantisch reicher Datenmodelle für die CDBS möglichen und in den lokalen Schemata enthaltenen, nutzerbezogenen Semantik. Diese nutzerbezogene Semantik ist oft nicht von allgemeinem Interesse für die Föderation.

Zur Umgehung der durch Konzeptreichtum und Komplexität verursachten Konflikte wird in [Sch95] die Verwendung eines semantisch armen Datenmodells, das auf dem relationalen Modell basiert und zusätzliche Konzepte wie Objektidentität [SS95] bietet, als kanonisches Datenmodell vorgeschlagen und diskutiert. Ein solches Datenmodell schränkt die Modellierungsfreiheiten für die Schematransformation ein. Durch semantische Reduktion werden die lokalen Schemata von nutzerspezifischer datenmodellabhängiger Semantik befreit. Die Menge der möglichen Konflikttypen für die resultierenden Komponentenschemata ist kleiner als bei Verwendung semantisch reicher Datenmodelle. Somit ist die Integration der Schemata einfacher, das resultierende integrierte Schema wird weniger komplex, wodurch sich die Verständlichkeit verbessert.

In den folgenden Abschnitten, die sich genauer mit der Modellhomogenisierung, der Schemaintegration und der Ableitung externer Schemata beschäftigen, wollen wir sowohl die Verwendung semantisch reicher, als auch die Verwendung semantisch armer Datenmodelle in unsere Überlegungen einbeziehen.

5.2 Datenmodellhomogenisierung

Die Überwindung der Datenmodellheterogenität der einzubeziehenden lokalen Schemata erfolgt durch eine Schematransformation vom Datenmodell des CDBS in das kanonische Datenmodell. Man kann die folgenden Arten lokaler Schemata unterscheiden:

- Ein lokales Schema im kanonischen Datenmodell oder einem dem kanonischem Datenmodell verwandten Datenmodell läßt sich direkt oder nur mit geringen Nachbesserungen als Komponentenschema verwenden. Die Funktionen zur Abbildung der

Daten sind nicht notwendig bzw. sehr einfach ableitbar.

- Steht das konzeptionelle Schema sowie Abbildungsfunktionen zwischen lokalem Schema und konzeptionellem Schema zur Verfügung, so kann das konzeptionelle Schema anstelle des lokalen Schemas als Komponentenschema genutzt werden. Dies ist insbesondere von Vorteil, wenn das konzeptionelle Schema auf einem dem kanonischen Datenmodell verwandten Datenmodell basiert.

- Besteht die Forderung nach Migration lokaler Applikationen auf die globale Schnittstelle des FDBS, so muß die nutzerspezifische Semantik während des Transformationsprozesses erhalten werden. Für die Verwendung eines semantisch armen Datenmodells als kanonisches Datenmodell schlägt [Sch95] die Nutzung spezieller Schemata zur Erhaltung der nicht auf das Komponentenschema zu übertragenden nutzerspezifischen Semantik vor.

- Bei der Einbeziehung von auf Dateien basierenden Datenhaltungssystemen ist eine systemspezifische nachträgliche Modellierung nötig. Diese Problemstellung kann abhängig von den bei der Datenhaltung angewandten Konzepten sehr aufwendig werden.

Ausgehend von diesen Klassifizierungsmöglichkeiten können Entscheidungen für das Vorgehen bei Schematransformation und Schemaintegration getroffen werden. Das grundlegende Ziel des modellhomogenisierenden Transformationsschrittes ist die Übersetzung des lokalen Schemas in ein äquivalentes Komponentenschema. Das Komponentenschema muß denselben Weltausschnitt beschreiben wie das lokale Schema. Diese Bedingung ist bei der Abbildung von semantisch reichen Datenmodellen auf semantisch arme Datenmodelle in der Regel nicht erfüllbar. Die Schematransformation ist somit mit einer Umstrukturierung verbunden, die gewisse Freiheiten erlaubt. Auch bei der Transformation von einem semantisch armen Datenmodell auf ein semantisch reiches Datenmodell bestehen Freiheiten bei der Modellierung. Diese Freiheiten können in Hinblick auf das Föderierungsziel genutzt werden.

Der wichtige Punkt der Einziehung von auf Dateisystemen basierender Datenhaltungssysteme bedarf der besonderen Untersuchung. Im allgemeinen kann man nicht von der Existenz eines auf einem etablierten Datenmodell beruhenden lokalen Schema sprechen, auch wenn die vom Systementwickler genutzten Standardmethoden zur Datenabspeicherung in gewisser Weise als Datenmodell angesehen werden können. Trotz des teilweise erfolgenden Einsatzes von Standarddateiformaten herrscht vor allem in der PC-Welt eine "babylonische Datenbeschreibungssprachverwirrung". Für derartige dateibasierte Systeme wird eine nachträgliche Modellierung des lokalen Schemas notwendig, die etwa der Schematransformation zugeordnet werden kann. Es ist günstig, zur nachträglichen Modellierung das kanonische Datenmodell zu nutzen, um eine nochmalige Transformation und evtl. damit verbundene Umstrukturierung zu vermeiden. Viele Probleme dieses Modellierungsschrittes haben ihre Ursache in der, in den meisten Datenhaltungssystemen nicht realisierten Trennung von konzeptioneller und interner Ebene bei gleichzeitiger Vermischung von Daten und Metadaten. Metadaten werden hierbei sowohl in den Dateien selbst, als auch in Dateinamen, der Hierarchie des Dateissystems und den Anwendungsprogrammen hinterlegt. Neben der Verfügbarkeit der Metadaten ist auch die Strukturierung der Dateien wichtig für die Transformation. Während einfache Standarddateiformate, die auf dem relationalen Ansatz verwandten Tabellen basieren (z.B. xBASE-Dateien) relativ einfach tranformierbar sind, existieren auch Formate, die dem NF^2-Modell ähnlich sind (z.B. die Arbeitspläne des Beispiels 1 aus Kapitel 3) und deren Transformation sich aufwendiger gestaltet. Andere Systeme (z.B. LAPLAS) nutzen textuelle Repräsentationen, in denen Daten und Metadaten unmittelbar verbunden sind. Die Abbildung der für spezielle Applikationen typischen Datenhaltungskonzepte erfordert einen erheblichen Aufwand, der aber durch die zusätzliche Funktionalität eines föderierten Datenbanksystems, wie Unterstützung des *Concurrent Engineering* und systemübergreifende Konsistenzsicherung zu rechtfertigen ist.

5.3 Schemaintegration

Der Prozeß der Schemaintegration umfaßt die Phasen Vorintegration, Erkennung und Lösung von Schemakonflikten sowie das Mischen und Restrukturieren [BLN86]. Ziel ist die Ableitung eines einheitlichen und weitgehend widerspruchsfreien föderierten Schemas. In der Phase der Vorintegration wird die Integrationsstrategie, insbesondere die Reihenfolge, in der die Schemata integriert werden sollen, bestimmt. Zu beachten ist, daß neben den Komponentenschemata der lokalen Datenbanken auch sich aus der Anforderungsanalyse des föderierten Datenbankentwurfs ergebende konzeptionelle Schemata zu integrieren sind.

Betrachten wir n-äre Strategien zur Schemaintegration, so ist ein zweistufiges Vorgehen unter Einsatz gewichteter und ungewichteter Verfahren möglich (Abbildung 5.1). Während bei ungewichteten Verfahren beide beteiligte Komponentenschemata im gleichen Maß umstrukturiert werden können, wird bei gewichteten Verfahren ein Komponentenschema dem anderen durch Umstrukturierung angeglichen, während das andere weitgehend erhalten bleibt. Wir können verschiedene Klassen zu integrierender Schemata unterscheiden und ihnen einen gewissen Grad der Umstrukturierbarkeit zuordnen, der als Wichtungsmaß für gewichtete Integrationsstrategien dienen kann.

- Aus der Anforderungsanalyse des FDBS abgeleitete konzeptionelle Schemata (Typ 1) können weitgehend umstrukturiert werden.
- Komponentenschemata, für die kein dem lokalen Schema äquivalentes externes Schema angeboten werden soll (Typ 2), sind eingeschränkt modifizierbar.
- Komponentenschemata, die nutzerspezifische Semantik enthalten, um äquivalente externe Schemata zu realisieren (Typ 3), sollten möglichst wenig umstrukturiert werden.

Ein weiteres Wichtungskriterium kann der Umfang und die Wichtigkeit der ererbten Datenbestände sein. In einer ersten Stufe der Schemintegration werden alle Schemata, die der selben Klasse zuzuordnen sind, mittels eines ungewichteten Verfahrens im gleichen Maße

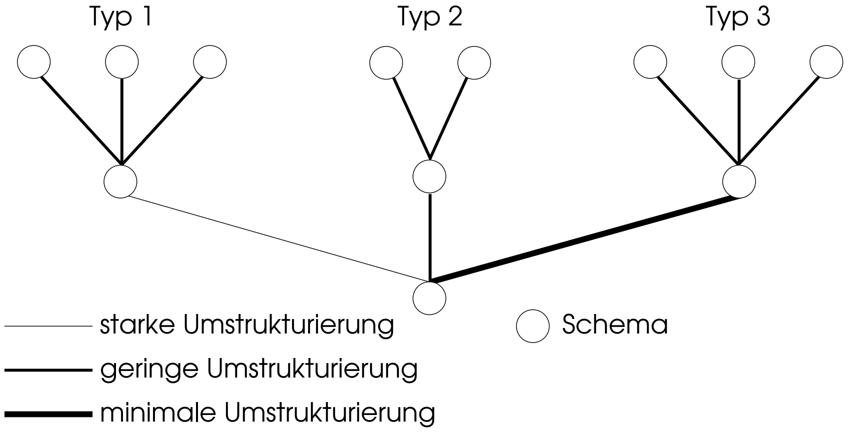

Abbildung 5.1: Beispiel für eine zweistufige Migrationsstrategie unter Nutzung gewichteter und ungewichteter Verfahren

umstrukturiert und integriert. In der zweiten Stufe werden die in der ersten Stufe abgeleiteten Zwischenschemata durch ein gewichtes Verfahren integriert. Auf dieses Weise bleibt möglichst viel der in den Typ 3-Schemata hinterlegten und zu bewahrenden Semantik erhalten. Das Vorgehen bei der Schemaintegration sollte anhand strategischer Überlegungen erfolgen, die sich am Gesamtprozeß des föderierten Datenbankentwurfs, also auch der Ableitung heterogener externer Schemata, der Datenintegration bei der Inbetriebnahme des FDBS und der Datenabbildung während des Betriebes, orientieren.

Ein wichtiges zu erforschendes Problem sind Methoden zur Erkennung und Lösung von Konflikten zwischen den zu integrierenden Schemata. Obwohl in einer Reihe von Veröffentlichungen die Behandlung spezieller Konfliktarten diskutiert wird, fehlt es noch an einer vollständigen Methodik. In [BLN86, SK93, KCGS95] werden Konflikte klassifiziert und beschrieben. In Abbildung 5.2 ist eine Klassifizierung nach [Hil95] dargestellt, die wir im folgenden kurz vorstellen werden. Auf eine ausführliche Diskussion von Methoden zur Erkennung und Lösung verzichten wir aus den genannten Gründen und da es den Rahmen dieser Arbeit sprengen würde.

Namenskonflikte sind Konflikte zwischen den Bezeichnungen von Schemaelementen unterschiedlicher Schemata. Sie werden weiter in *Homonyme* (gleiche Namen für semantisch nicht vergleichbare Schemaelemente) und *Synonyme* (unterschiedliche Bezeichnung für semantisch vergleichbare Schemaelemente) unterschieden.

Semantische Konflikte bestehen zwischen semantisch vergleichbaren Schemaelementen, wenn die zugehörigen Realweltklassen [SPD92] nicht übereinstimmen. Bei den semantischen Konflikten wird weiterhin in *Inklusion*, *Überlappung* und *Disjunkheit* der zugehörigen Realweltklassen unterschieden. Ein Beispiel für Überlappung ist der Konflikt zwischen den möglichen Extensionen der Klassen Person in der Datenbank der Universität und in der Datenbank des Einwohnermeldeamtes der Universitätsstadt.

Beschreibungskonflike haben ihre Ursache in der intensionalen Verschiedenheit semantisch vergleichbarer Objekttypen. Sie resultieren zum einen aus der unterschiedlichen Weltsicht der Modellierer, zum anderen aus der durch unterschiedlich reiche Datenmodel-

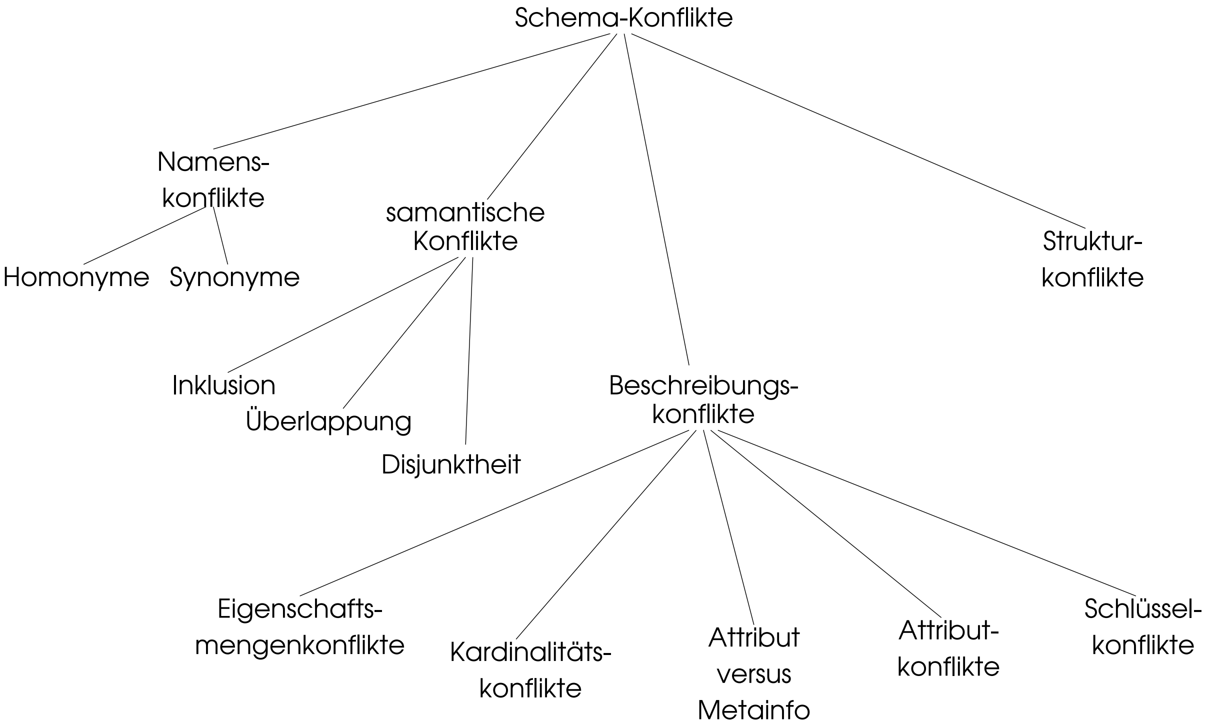

Abbildung 5.2: Schemakonflikte nach [Hil95]

le zur Verfügung gestellte Darstellungsmittel. Eine weitere Unterteilung in *Eigenschaftsmengenkonflikte*, *Attributkonflikte*, *Kardinalitätskonflikte* zwischen *Beziehungen*, *Attribut vs. Metainfo* und *Schlüsselkonflikte* unterstützt die Erkennung und Lösung von Beschreibungskonflikten.

Strukturkonflikte ergeben sich durch die Nutzung unterschiedlicher Strukturelemente zur Abbildung äquivalenter Realweltkonzepte. Mögliche Strukturkonflikte sind durch die Menge der durch das genutzte (kanonische) Datenmodell zur Verfügung gestellten Strukturelemente gegeben.

Der Prozeß der schrittweisen Lösung erkannter Konflikte kann als Normalisierung der Schemata betrachtet werden. Konfliktfreie Schemata werden durch Umstrukturierung abgeleitet. Dabei ist eine gewichtete Umstrukturierung beteiligter Schemata zur weitgehenden Erhaltung benötigter Semantik hilfreich.

Das Auftreten der meisten Konfliktarten ist nicht an bestimmte Datenmodelle gebunden. Allerdings kann die, durch die bei der Modellierung mittels semantisch reicher Datenmodelle bedingte, nutzerspezifische Struktur der Komponentenschemata zu einer großen Zahl einzelner Konflikte führen. Auch aus diesen Gründen wird durch [Sch95] die Verwendung semantisch armer Datenmodelle favorisiert.

Nach der Erkennung und Beseitigung der Schemakonflikte kann auf Basis der normalisierten Schemata durch Mischung ein gemeinsames integriertes Schema abgeleitet werden. Dieses Schema kann wiederum an weiteren Normalisierungs- und Integrationsschritten mit anderen Schemata beteiligt sein. Sind alle Komponentenschemata in einem

Schema integriert, so erhält man das globale, föderierte Schema.

Im allgemeinen lassen sich nicht alle Konflikte vollständig lösen. Dashalb sind bei der Schemaintegration Kompromisse zwischen der Vollständigkeit des föderierten Schemas einerseits und der Durchführbarkeit der Integration andererseits, zu schliessen. Diese Kompromisse werden durch spezifische Rahmenbedingungen des Föderierungsszenarios beeinflußt.

5.4 Heterogenisierung

Nicht immer entspricht das föderierte Schema den Anforderungen der Applikationen, die über die globale Schnittstelle auf das föderierte Datenbanksystem zugreifen möchten. Insbesondere ist das der Fall, wenn eine Migration lokaler Applikationen vom lokalen Komponentendatenbanksystem auf das föderierte Datenbanksystem vorgesehen ist. In diesem Fall ist es notwendig, externe Schemata anzubieten, die äquivalent zu den lokalen Schemata sind. Durch eine erneute Schematransformation kann für spezifische Applikationen bzw. Nutzer sowohl Schema- als auch Datenmodellheterogenität hergestellt werden. Die dazu benötigte nutzerspezifische Semantik kann bei semantisch reichen Datenmodellen aus dem föderierten Schema übernommen werden, falls sie nicht durch den Umstrukturierungsprozeß während der Integration verlorengegangen ist. Bei Verwendung eines semantisch armen Datenmodells als kanonisches Datenmodell muß diese Semantik in speziellen Schemata erhalten werden, mit denen eine semantische Anreicherung während des Heterogenisierungsprozesses durchgeführt werden kann [Sch95].

5.5 Zusammenfassung

Der Prozeß des föderierten Datenbankentwurfs umfaßt die Phasen der modellhomogenisierenden Schematransformation, der schemahomogenisierenden Schemaintegration und der heterogenisierenden Schematransformation. Diese Themenbereiche bedürfen wegen des Fehlens einer theoretisch basierten und praktisch anwendbaren Methodik weiterer Untersuchungen. Dabei sind sowohl semantisch reiche als auch semantisch arme kanonische Datenmodelle zu untersuchen. Von besonderer Bedeutung für das Projekt **SIGMA**$_{FDB}$ ist die Analyse und Aufarbeitung von Verfahren zur Transformation und Modellierung von Schemata, die in dateibasierten Datenhaltungssystemen implizit enthalten sind.

Kapitel 6

Konsistenzsicherung in FDBS

Ein wesentlicher Aspekt beim Zusammenschluß heterogener Systeme ist die Konsistenz der föderierten Daten. Obwohl die Föderierung von autonomen heterogenen (Datenbank-) Systemen Thema vieler Veröffentlichungen war, ist die systemübergreifende Konsistenzsicherung in solchen Systemen bislang kaum behandelt worden.

Aktive Mechanismen zur Konsistenzsicherung in aktiven Datenbanken (z.B. REACH [BZBW95], Sentinel [CKTB95], SAMOS [GGD94]) stellen ein aktuelles und vielversprechendes Prinzip zur Konsistenzsicherung in einzelnen, aber auch in föderierten Datenbanksystemen dar. Sie beruhen auf regelbasierter Konsistenzüberwachung und -erzwingung. Für potentiell konsistenzverletzende Aktionen werden Regeln definiert, die die für die jeweilige Situation geeigneten Korrekturmaßnahmen festlegen. Tritt eine solche Aktion (d.h. ein Ereignis) ein, wird die entsprechende Regel aktiviert, die dann die konsistenzwahrenden Maßnahmen initiiert (falls erforderlich). So ist es möglich, eine regelorientierte Wissensbasis zur Konsistenzsicherung des gesamten Datenbestandes anzulegen.

Die bislang erforschten Prinzipien aktiver Datenbanken beziehen sich ausschließlich auf einzelne Datenbanksysteme oder aber zumindest auf homogene Datenbanksysteme. Ein Ziel des Projektes **SIGMA**$_{FDB}$ ist es, diese Prinzipien im Kontext föderierter, heterogener Datenbanksysteme zu analysieren und eventuelle Erweiterungen aktiver Mechanismen zu betrachten. Da die kommerziellen Datenbanksysteme in der Regel keine aktiven Mechanismen zur Konsistenzsicherung anbieten, sind die aktiven Mechanismen auf der Ebene des föderierten Systems, also datenbankübergreifend anzusiedeln. Fast alle kommerziellen Datenbanksysteme bieten jedoch eigene Integritätsüberwachungsmechanismen an, die aber nur einen Bruchteil der Möglichkeiten aktiver, regelbasierter Mechanismen leisten. Aus Effizienzgründen sollten diese mitgelieferten Mechanismen (wenn möglich) miteinbezogen werden. Angestrebt wird dabei eine Aufteilung derart, daß die aktiven, regelbasierten Mechanismen die globale Kontrolle bei der Konsistenzsicherung übernehmen und die lokal zu behandelnden Teile an die einzelnen Datenbanksysteme weitergeben.

Zur Realisierung eines solchen Konzeptes sind zunächst geeignete (erweiterte) Konzepte zur Spezifikation und Verarbeitung von datenbankübergreifenden (Integritäts-) Regeln für föderierte Datenbankumgebungen abzuleiten bzw. weiterzuentwickeln. Dazu werden wir im folgenden Abschnitt die ECA-Regel-Spezifikation in föderierten Systemen diskutieren. Zum Abschluß werden wir dann in Abschnitt 6.2 einen ersten Ansatz für eine aktive föderierte Datenbankarchitektur skizzieren.

6.1 ECA-Regel-Spezifikation in FDBS

Als Beschreibungsmittel von aktiven Eigenschaften haben sich die *ECA-Regeln* [DBB+88, DHW95] als de facto Standard durchgesetzt. Wir möchten ECA-Regeln dazu verwenden, auf der Föderierungsebene potentiell konsistenzverletzende Ereignisse, auf die das System mit entsprechenden konsistenzwahrenden Aktionen reagieren soll, zu spezifizieren (siehe unten).

define rule *Globale_Integritätsregel*
on *potentiell konsistenzverletzendes Ereignis*
if *globale Bedingung formuliert über den FDBS-Zustand*
do *globale Aktion(en) zur Erhaltung bzw. Wiederherstellung der FDBS-Konsistenz*

ECA-Regeln scheinen aufgrund ihrer Mächtigkeit und Flexibilität ein geeigneter Mechanismus zur Konsistenzsicherung auch für föderierte Datenbankumgebungen zu sein. Die allgemein zu beantwortende Frage ist, wie mächtig darf bzw. soll die zu verwendende ECA-Regelsprache sein, um als geeignetes Mittel zur Spezifikation von Integritätsregeln in solchen Umgebungen dienen zu können?

Ereignisteil: Gemäß [GD94, CM93] muß ein aktives System die Spezifikation und Detektion von diversen Ereignistypen, wie z.B. Zeitereignisse, Methodenaufrufe (Operationsaufrufe) oder zusammengesetzte Ereignisse [GJS92, CKAK94] unterstützen. Für föderierte Umgebungen ergeben sich nun die folgenden Fragestellungen, die zu lösen sind:

- Welche der vorgeschlagenen Ereignistypen sollten bzw. müssen von einem föderierten System für Konsistenzsicherungszwecke unterstützt werden? Sind Erweiterungen nötig?
- Wie können Ereignisse in CDBS erkannt werden? In aktiven objektorientierten Datenbanksystemen wird hierzu das Method Wrapping benutzt (vgl. [BZBW95]). Doch wie können Ereignisse in relationalen Datenbanksystemen erkannt und signalisiert werden? Eine naheliegende Lösung ist, Trigger (als lokale Regelmanager) dazu zu verwenden, das Eintreten eines (lokalen) Ereignisses anzuzeigen. Falls aber keine Trigger unterstützt werden, muß ebenfalls eine Art "Wrapper" in das System integriert werden, so daß Ereignisse vor bzw. nach dem Aufruf einer SQL-Operation signalisiert werden können. Wenn bestehende, nicht mit aktiven Fähigkeiten (z.B. mit Triggermechanismen) versehene CDBS föderiert werden, entsteht das Problem, daß weder die CDBS selbst noch ihre Anwendungen Ereignisse erkennen und signalisieren können. In diesen Fällen müssen die Systeme (inkl. Anwendungen) "gewrapped", d.h. neu übersetzt werden. Dies steht jedoch im starken Widerspruch zur Forderung nach lokaler Autonomie.

- Wie können Ereignisse, die in den CDBS erkannt werden, der Föderierungsschicht bekannt gemacht werden? Hier müssen *lokale* ECA-Regeln, die die Detektion der lokalen Ereignisse an die entsprechende regel- bzw. ereignisverarbeitende Komponente der Föderationsschicht (d.h. einem globalen Regelmanager) weiterleiten, von der globalen ECA-Regeldefinition abgeleitet werden. Für die Erzeugung von lokalen ECA-Regeln sind entsprechende Kooperationsmechanismen zwischen lokalen und globalen Regelmanagern erforderlich.

Bedingungsteil: Welche Arten von Bedingungen können bzw. dürfen formuliert werden? Wünschenswert sind hier Bedingungen, die über die Zustände der diversen CDBS spezifiziert werden können.

Aktionsteil: Welche Aktionen sind zulässig? Beliebige Methoden- bzw. Operationsaufrufe sollen als Aktionen möglich sein, d.h. sowohl benutzerdefinierte Methoden als auch Systemoperationen, wie z.B. SQL-Anweisungen (insert, update, delete, etc.), sollen ausführbar sein. Weiterhin sollen vordefinierte Transaktionen/Applikationen der jeweiligen CDBS als zulässige Sequenzen von Aktionen genutzt werden können.

Es muß ebenfalls möglich sein, systemübergreifende Aktionen aus einer ECA-Regel heraus zu starten, d.h. es muß die Möglichkeit bestehen, mehrere Methoden (bzw. Operationen) auf verschiedenen CDBS aufrufen zu können. Hierfür müssen jedoch geeignete Synchronisationsmechanismen vom föderierten System bereitgestellt werden.

Kopplungsmodi: Ein wesentlicher Flexibilitätsfaktor für aktive Systeme sind die bereitgestellten Kopplungsmodi. Entsprechend [BBKZ94] muß ein aktives System diverse Kopplungsmodi unterstützen. Die Kopplungsmodi bestimmen, wann und in welchen Transaktionsgrenzen die Aktionen einer ECA-Regel ausgeführt werden. Hier ist die Frage zu klären, welche der vorgeschlagenen Kopplungsmodi im Rahmen einer föderierten Umgebung für die Konsistenzsicherung notwendig bzw. geeignet sind. Als Grundlage für die Realisierung diverser Kopplungsmodi wird allerdings die Existenz eines föderierten Transaktionsmechanismus [BGS92, BÖH$^+$92, SSW95] vorausgesetzt. Dieser Mechanismus sollte zumindestens geschachtelte Transaktionen unterstützen, damit systemübergreifende Aktionen in Subtransaktionen innerhalb einer globalen Transaktion ablaufen können.

6.2 Integration aktiver Mechanismen in föderierte Datenbanksysteme

Aufbauend auf der in Kapitel 2.2 skizzierten Architektur verfolgen wir in diesem Abschnitt den Ansatz, aktive Mechanismen zur systemübergreifenden Konsistenzsicherung von föderierten Daten auf der Föderierungsebene zu integrieren. Unser Ziel im Rahmen des **SIGMA**$_{FDB}$-Projektes ist es, die von den Einzelsystemen bereitgestellten Integritätmechanismen zur *globalen* Konsistenzsicherung heranzuziehen.

Die Integration des ECA-Regelkonzeptes in die in Abbildung 2.4 skizzierte föderierte Umgebung kann wie folgt erfolgen. Auf der Föderierungsebene werden globale Integritätsmechanismen in Form von *globalen* ECA-Regeln bereitgestellt (siehe Abbildung 6.1). Diese

ECA-Regeln werden in der globalen Datenbank (in der sich die Daten der Föderierungsschicht befinden) abgelegt.

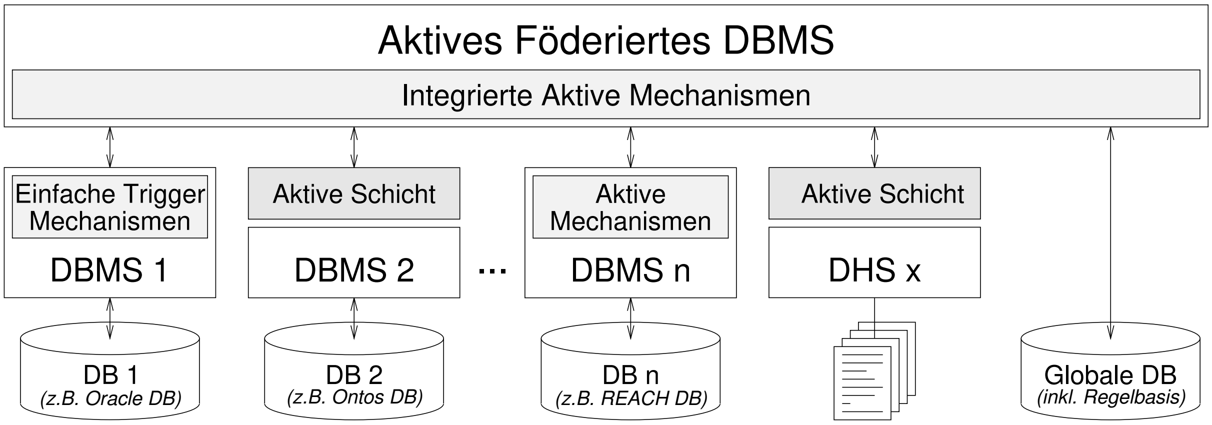

Abbildung 6.1: Aktive Föderierte Datenbank Architektur

Das Erkennen von Ereignissen, die sich ausschließlich auf die globale Datenbank beziehen, kann bereits auf der Föderierungsebene erfolgen, während für die Detektion der übrigen Ereignisse die Kooperation/Kommunikation mit den Komponentensystemen erforderlich ist. In diesen Fällen müssen die Ereignisse lokal von den einzelnen Systemen (*durch lokale Regelmanager*) erkannt und an die entsprechende Komponente der Föderierungsschicht (*einen globalen Regelmanager*) weitergeleitet werden. Dieser Regelmanager ist dann für die Komposition von *zusammengesetzten* Ereignissen [CM93] sowie für die Verarbeitung der entsprechenden Regeln verantwortlich. Hierzu muß der globale Regelmanager zunächst Maßnahmen zur Auswertung der spezifizierten Bedingungen einleiten und im Falle einer erfolgreichen Bedingungsauswertung die notwendigen Aktionen starten, indem er entweder Operationen (Funktionen bzw. Methoden) auf der globalen Datenbank ausführt oder entsprechende Operationen der Komponentensysteme aufruft.

Um die in Abbildung 6.1 skizzierte Architektur zu realisieren, müssen wir aktive Mechanismen sowohl auf der globalen wie auch auf der lokalen Ebene anbieten. Hier ist zu beachten, daß die einzelnen Komponentensysteme zum Teil schon aktive Mechanismen unterstützen. In diesem Fall ist zu überprüfen, ob diese Mechanismen (z.B. Trigger relationaler Datenbanksysteme) für unsere Zwecke eingesetzt werden können. Für den Fall, daß keinerlei aktive Mechanismen durch das lokale Komponentensystem (z.B. traditionelles dateibasiertes Datenhaltungssystem) unterstützt werden, muß eine sogenannte "aktive Schicht" für das Komponentensystem bereitgestellt werden. Alle Operationsanforderungen an dieses Komponentensystem müssen dann diese Schicht passieren. In dieser Schicht können die aktiven Mechanismen derart realisiert werden, daß (lokale) Ereignisse erkannt und verarbeitet werden. Dies impliziert jedoch, daß alle lokalen Anfragen über diese aktive Schicht "umgeleitet" werden. Eine detailliertere Beschreibung der aktiven föderierten Datenbankrahmenarchitektur kann in [CT95] nachgelesen werden.

Als abschließende Bemerkung möchten wir festhalten, daß die Wahrung der lokalen Autonomie der Komponentensysteme und die globale Konsistenzsicherung von föderierten

Daten zwei gegensätzliche Anforderungen darstellen. Eine globale Konsistenzsicherung ist ohne Einschränkungen der lokalen Autonomie nicht geeignet realisierbar. Darüber hinaus sind wir der Meinung, daß globale, systemübergreifende Konsistenzsicherung nur in eng gekoppelten Föderationen, in denen die beteiligten heterogenen Datenbestände in hohem Maße korrelieren, sinnvoll bzw. erwünscht sind.

Kapitel 7

Zusammenfassung und Ausblick

Dieser Bericht dient der Zusammenfassung wichtiger Grundlagen und erster Ideen bei der Bearbeitung des Projektes **SIGMA**$_{FDB}$, das im Rahmen des Forschungsschwerpunktes "Neue Produktionssysteme/Experimentelle Fabrik" durch das Land Sachsen-Anhalt gefördert wird. Die unter dieser Überschrift begonnene interdisziplinäre Zusammenarbeit von Maschinenbauern und Informatikern ermöglicht die Untersuchung des Einsatzes neuer Techniken und Methoden für das integrierte Arbeiten während des Fabrikplanungsprozesses. In diesem Sinne dokumentieren die Beispiele aus dem Anwendungsgebiet Fabrikplanung Möglichkeiten des Einsatzes föderierter Datenbanksysteme. Bei der Vorstellung wesentlicher Standards und prototypischer Systeme wurde deren mögliche Bedeutung für das Projekt **SIGMA**$_{FDB}$ diskutiert.

Folgende, für die weitere Arbeit wichtige Erkenntnisse ergeben sich für die Erforschung und den Einsatz föderierter Datenbanksysteme für die Fabrikplanung:

- Zentrale Bedeutung hat die integrierte Nutzung der Daten vorhandener Softwarewerkzeuge bei Wahrung der lokalen Autonomie.

- Die konzeptionelle Verbesserung bestehender Systeme kann durch Unterstützung der sanften Migration von Daten und Applikationen durch FDBS erfolgen. Aus diesem Grund ist die Bereitstellung von zu lokalen Schemata äquivalenten externen Schemata konzeptionell zu untersuchen.

- Beim Entwurf des FDBS sind verschiedene Datenmodelle zu unterstützen.

- Eine Reihe neuerer PC-Applikationen können über ODBC-kompatible Schnittstellen auf Datenbanksysteme zugreifen. Aus diesem Grunde ist die Bereitstellung externer Schnittstellen, die den ODBC-Standard unterstützen, zu untersuchen.

- Die Einbeziehung lokaler Datenhaltungssysteme, die auf Dateien basieren, ist wichtig. Für die deshalb durchzuführende nachträgliche Modellierung (Schematransformation) sind Konzepte und Methoden zu erforschen.

- Bei der integrierten Nutzung von Systemen müssen globale Mechanismen zur Konsistenzsicherung eingesetzt werden, die die Redundanz der föderierten Datenbestände kontrollieren.
- Zur Offenlegung lokaler Transaktionen kommt in einigen kommerziellen Datenbanksystemen das XA-Protokoll zum Einsatz. Durch dessen Ausnutzung kann der Föderierungsdienst die lokalen Mechanismen zur globalen Transaktionsverwaltung mitbenutzen.
- In einer Reihe lokaler Datenhaltungssysteme existieren keine Mechanismen zur Konsistenzsicherung und zur Unterstützung des Mehrbenutzerbetriebs. Diese Funktionalität kann durch globale Mechanismen, die Teil des Föderierungsdienstes sind, nachgebildet werden.

Sowohl im theoretisch-konzeptionellen als auch im praktisch-technischen Bereich besteht auf dem Gebiet föderierter Datenbanksysteme Forschungsbedarf. Bei der Validierung gewonnener Erkenntnisse in Zusammenarbeit mit Partnern aus dem Maschinenbau können allgemeine und spezielle Problemstellungen realitätsnah untersucht werden.

Literaturverzeichnis

[ABD+89] M. Atkinson, F. Bancilhon, D. DeWitt, K. Dittrich, D. Maier und S. Zdonik. The Object-Oriented Database System Manifesto. In W. Kim, J. M. Nicolas und S. Nishio, Herausgeber, *Proc. of the 1st Int. Conf. on Deductive and Object Oriented Databases, Kyoto, Japan*, S. 40–57. North Holland, Dezember 1989.

[Agg90] B. Aggteleky. *Fabrikplanung*, Band 1–3. C. Hanser Verlag, München-Wien, 1990.

[BBKZ94] H. Branding, A. Buchmann, T. Kudrass und J. Zimmermann. Rules in an Open System: The REACH Rule System. In N. W. Paton und M. H. Williams, Herausgeber, *Proc. of the 1st Int. Workshop on Rules in Database Systems (RIDS'93), Edinburgh, Scotland, August 1993*, S. 111–126. Workshops in Computing, Springer-Verlag, 1994.

[BEK+94] R. Böttger, Y. Engel, G. Kachel, S. Kolmschlag, D. Nolte und E. Radeke. Enhancing the Data Openness of Framworks by Database Federation Services. Technical Report, Cadlab Paderborn, Germany, 1994.

[BFHK94] R. Busse, P. Fankhauser, G Huck und W. Klas. IRO-DB: An Object-Oriented Approach Towards Federated and Interoperable DBMS. In *Proc. of the Int. Workshop on Advances in Databases and Information Systems (ADBIS'94), Moscow, Russia*, S. 178–186. Russian Academy of Sciences, Mai 1994.

[BFN94] R. Busse, P. Fankhauser und E. J. Neuhold. Federated Schemata in ODMG. In J. Eder und L. A. Kalinichenko, Herausgeber, *Extending Information Systems Technology – Proc. of the 2nd Int. East/West Database Workshop, Klagenfurt, Austria*, S. 356–379. Workshops in Computing, Springer-Verlag, September 1994.

[BG95] M. Bounab und C. Godart. A Federated Approach to Tool Integration. In J. Iivari, K. Lyytinen und M. Rossi, Herausgeber, *Proc. of the 7th Conf. on Advanced Information System Engineering (CAiSE'95), Jyvaskyla, Finland*, S. 269–282. LNCS 932, Springer-Verlag, Juni 1995.

[BGS92] Y. Breitbart, H. Garcia-Molina und A. Silberschatz. Overview of Multidatabase Transaction Management. *The VLDB Journal*, 1(2):181–240, Oktober 1992.

[BHP92] M. Bright, A. Hurson und S. Pakzad. A Taxonomy and Current Issues in Multidatabase Systems. *IEEE Computer*, 25(3):50–56, März 1992.

[BK95] R. Böttger und G. Kachel. Implementierungsaspekte der Database Federation Services. In C. Eckert, H.-J. Klein und T. Polle, Herausgeber, *7. GI-Workshop "Grundlagen von Datenbanken"*, S. 25–29. Hildesheimer Informatik Berichte, Juni 1995.

[Bla95] A. Blaser. Die BTW im Wandel der Datenbank-Zeiten. In G. Lausen, Herausgeber, *Proc. GI-Fachtagung "Datenbanksysteme in Büro, Technik und Wissenschaft" (BTW'95), Dresden* , S. 48–50. Informatik aktuell, Springer-Verlag, Mai 1995.

[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.

[BÖH+92] A. Buchmann, M. T. Özsu, M. Hornick, D. Georgakopoulos und F. Manola. A Transaction Model for Active Distributed Object Systems. In A. K. Elmagarmid, Herausgeber, *Database Transaction Models for Advanced Applications*, S. 123–151. Morgan Kaufman Publishers, 1992.

[BS95] M. L. Brodie und M. Stonebraker, Herausgeber. *Migrating Legacy Systems*. Morgan Kaufmann Publishers, 1995.

[BZBW95] A. P. Buchmann, J. Zimmermann, J. A. Blakeley und D. L. Wells. Building an Integrated Active OODBMS: Requirements, Architecture, and Design Decisions. In P. S. Yu und A. L. P. Chen, Herausgeber, *Proc. of the 11th IEEE Int. Conf. on Data Engineering (ICDE'95), Taipei, Taiwan*, S. 117–128. IEEE Computer Society Press, März 1995.

[Cas93] M. Castellanos. Semantic Enrichment of Interoperable Databases. In H.-J. Schek, A. P. Sheth und B. D. Czejdo, Herausgeber, *Proc. of the 3rd Int. Workshop on Research Issues in Data Engineering: Interoperability in Multidatabase Systems (RIDE-IMS'93), Vienna, Austria*, S. 126–129. IEEE Computer Society Press, April 1993.

[Cat94] R. G. Cattell, Herausgeber. *The Object Database Standard: ODMG-93*. Morgan Kaufmann Publishers, San Mateo, CA, 1994.

[CGW94] S. Chawathe, H. Garcia-Molina und J. Widom. Constraint Management in Loosely Coupled Distributed Databases. Technical Report, Department of Computer Science, Stanford University, 1994.

[CKAK94] S. Chakravarthy, V. Krishnaprasad, E. Anwar und S.-K. Kim. Composite Events for Active Databases: Semantics, Contexts and Detection. In J. B. Bocca, M. Jarke und C. Zaniolo, Herausgeber, *Proc. of the 20th Int. Conf. on Very Large Data Bases (VLDB'94), Santiago, Chile*, S. 606–617. Morgan Kaufmann Publishers, 1994.

[CKTB95] S. Chakravarthy, V. Krishnaprasad, Z. Tamizuddin und R. H. Badani. ECA Rule Integration into an OODBMS: Architecture and Implementation. In P. S. Yu und A. L. P. Chen, Herausgeber, *Proc. of the 11th IEEE Int. Conf. on Data Engineering (ICDE'95), Taipei, Taiwan*, S. 341–348. IEEE Computer Society Press, März 1995.

[CM93] S. Chakravarthy und D. Mishra. Snoop: An Expressive Event Specification Language for Active Databases. Technischer Bericht UF-CIS-TR-93-007, Computer Science Department, University of Florida, März 1993.

[Cod82] E. F. Codd. Relational Database: A Practical Foundation for Productivity. *Communications of the ACM*, 25(2):109–117, Februar 1982.

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

[DAF86] Database Architecture Framework Task Group (DAFTG) of the ANSI/X3/SPARC Database System Study Group DAFTG. Reference Model for DBMS Standardization. *ACM SIGMOD RECORD*, 15(1):19–58, 1986.

[Dat87] C. J. Date. *A Guide to THE SQL STANDARD*. Addison-Wesley, Reading, MA, 3 Auflage, November 1987.

[DBB+88] U. Dayal, B. Blaustein, A. Buchmann, S. Chakravarthy, D. Goldhirsch, M. Hsu, R. Ladin, D. McCarthy und A. Rosenthal. The HiPAC Project: Combining Active Databases and Timing Contraints. *ACM SIGMOD RECORD*, 17(1):51–70, März 1988.

[DD93] C. J. Date und H. Darwen. *A Guide to the SQL Standard*. Addison-Wesley, Reading, MA, 1993.

[DHW95] U. Dayal, E. Hanson und J. Widom. Active Database Systems. In W. Kim, Herausgeber, *Modern Database Systems*, S. 434–456. ACM Press, 1995.

[Dup94] Y. Dupont. Resolving Fragmentation Conflicts in Schema Integration. In P. Loucopoulos, Herausgeber, *Entity-Relationship Approach — ER'94 (Proc. of the 13th Int. Conf. on the Entity-Relationship Approach, Manchester, UK)*, S. 513–532. LNCS 881, Springer-Verlag, Dezember 1994.

[EHSS91] M. Eigner, C. Hiller, S. Schindewolf und M. Schmich. *Engineering Database*. C. Hanser Verlag, München-Wien, 1991.

[FS94] R. Fietz und C. Schmidt. Weiterentwicklung des Arbeitsplatzes “Logistikgerechte Fabrikplanung”. In *Jahresbericht 1994*, S. 42–45. Fraunhoferinstitut für Fabrikbetrieb- und automatisierung Magdeburg, 1994.

[FS96] R. Fietz und C. Schmidt. LOGAP - Ein entwicklungsfähiges Konzept für die Fabrikplanung der Zukunft. *CIM Management*, 11(3), März 1996.

[GAP93] H. Grabowski, R. Anderl und A. Polly. *Integriertes Produktmodell*. Beuth Verlag GmbH, 1993.

[GD94] S. Gatziu und K. R. Dittrich. Events in an Active Object-Oriented Database System. In N. W. Paton und M. H. Williams, Herausgeber, *Proc. of the 1st Int. Workshop on Rules in Database Systems (RIDS'93), Edinburgh, Scotland, August 1993*, S. 23–39. Workshops in Computing, Springer-Verlag, 1994.

[GGD94] S. Gatziu, A. Geppert und K. R. Dittrich. SAMOS: an Active Object-Oriented Database System. Technical Report 94.16, University of Zurich, 1994.

[GGF+95] Gardarin. G., S. Gannouni, B. Finance, P. Fankhauser, W. Klas, D. Pastre, R. Legoff und A. Ramfos. IRO-DB — A Distributed System Federating Object and Relational Databases. In O. Bukhres und A. K. Elmagarmid, Herausgeber, *Object Oriented Multidatabase Systems – A Solution for Advanced Applications*, Kapitel 20. Prentice Hall, 1995.

[GJS92] N. H. Gehani, H. V. Jagadish und O. Shmueli. Composite Event Specification in Active Databases. In L.-Y. Yuan, Herausgeber, *Proc. of the 18th Int. Conf. on Very Large Data Bases (VLDB'92), Vancouver, Canada*, S. 265–276. Morgan Kaufmann Publishers, August 1992.

[Gro91] Object Management Group. *The Common Object Request Broker: Architecture and Specification.* OMG Document Number 91.12.1, Dezember 1991.

[Gro94] Object Management Group. *Common Object Services Specification, Volume 1.* OMG Document Number 94.3.1, März 1994.

[Gro95] Objekt Management Group. *OMG's Object Models.* OMG Document Number 95.3.22, März 1995.

[GSCS95] M. Garcia-Solaco, M. Castellanos und F. Saltor. A Semantic-Discriminated Approach to Integration in Federated Databases. In S. Laufmann, S. Spaccapietra und T. Yokoi, Herausgeber, *Proc. of the 3rd Int. Conf. on Cooperative Information Systems (CoopIS'95), Vienna, Austria*, S. 19–31, Mai 1995.

[Här94] M. Härtig. *Objektorientierte Integration von autonomen Datenhaltungssystemen.* Dissertation, Verlag Dr. Kovač, Hamburg, 1994.

[HD93] M. Härtig und K. Dittrich. An Object-Oriented Integration Framework for Building Heterogeneous Database Systems. In D. K. Hsiao, E. J. Neuhold und R. Sacks-Davis, Herausgeber, *Interoperable Database Systems (DS-5), Proc. of the IFIP WG 2.6 Database Semantics Conf., Lorne, Victoria, Australia, November, 1992*, S. 33–54. North-Holland, 1993.

[Hil95] E. Hildebrandt. Eignungsuntersuchung des ODMG-Objektmodells als Objektmodell einer Föderierungsschicht zur Integration heterogener, autonomer Datenbanksysteme. Diplomarbeit, Otto-von-Guericke-Universität Magdeburg, Fakultät für Informatik, Oktober 1995.

[HK95] U. Hohenstein und C. Körner. Semantische Anreicherung relationaler Datenbanken. In G. Lausen, Herausgeber, *Proc. GI-Fachtagung "Datenbanksysteme in Büro, Technik und Wissenschaft" (BTW'95), Dresden*, S. 130–149. Informatik aktuell, Springer-Verlag, Mai 1995.

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

[Hoh95] U. Hohenstein. Query Processing in Semantically Enriched Relational Databases. In *Basque Int. Workshop on Information Technology: Data Management Systems (BIWIT'94), San Sebastian, Spain*. IEEE Computer Society Press, 1995.

[HS95] A. Heuer und G. Saake. *Datenbanken — Konzepte und Sprachen*. International Thomson Publishing, Bonn, 1995.

[Hsi92a] D. Hsiao. Federated Databases and Systems: Part I – A Tutorial on Their Data Sharing. *The VLDB Journal*, 1(1):127–179, Juli 1992.

[Hsi92b] D. Hsiao. Federated Databases and Systems (Part II) – A Tutorial on Their Resource Consolidation. *The VLDB Journal*, 1(2):285–310, Oktober 1992.

[Jan95] S. Janssen. Realisierung einer Anfragesprache für objektorientiert remodellierte relationale Datenbestände. Diplomarbeit, Otto-von-Guericke-Universität Magdeburg, Fakultät für Informatik, November 1995.

[Kac94] G. Kachel. Coupling of Heterogeneous Databases by Federation Services. Technical Report, Cadlab Paderborn, September 1994.

[Kac95] G. Kachel. Unterstützung der 5-Ebenen-Schemaarchitektur durch Kontexte. In C. Eckert, H.-J. Klein und T. Polle, Herausgeber, *7. GI-Workshop "Grundlagen von Datenbanken"*, S. 63–67. Hildesheimer Informatik Berichte, Juni 1995.

[KBB+95] S. Kolmschlag, R. Böttger, B. Burkert, Y. Engel, G. Kachel, D. Nolte und E. Radeke. Überblick zum Projekt "DBF - Database Federation Services". In C. Eckert, H.-J. Klein und T. Polle, Herausgeber, *7. GI-Workshop "Grundlagen von Datenbanken"*, S. 78–82. Hildesheimer Informatik Berichte, Juni 1995.

[KCGS95] W. Kim, I. Choi, S. Gala und M. Scheevel. On Resolving Schematic Heterogeneity in Multidatabase Systems. In W. Kim, Herausgeber, *Modern Database Systems*, S. 318–337. ACM Press, 1995.

[Klu95] M. Klusch. Towards a Federative System FCSI for a Contextual Recognition of Interdatabase Dependencies. In *Proc. of the 18. Int. Conf. ETCE-95, Houston, TX, USA*, S. 233–240, Februar 1995.

[Lam94] C. Lambert. *ODBC: Architecture, Performance and Tuning*. Microsoft ODBC Product Group, März 1994.

[LMR90] W. Litwin, L. Mark und N. Roussopoulos. Interoperability of Multiple Autonomous Databases. *ACM Computing Surveys*, 22(3):267–293, September 1990.

[MKN90] P. Muth, W. Klas und E. Neuhold. How to Handle Global Transactions in Heterogeneous Database Systems, November 1990.

[MRB+92] S. Mehrotra, R. Rastogi, Y. Breitbart, H. F. Korth und A. Silberschatz. Efficient Global Transaction Management in Multidatabase Systems. Technischer Bericht CS-TR-92-11, University of Texas at Austin, Department of Computer Sciences, 1992.

[MSJ94] H. Mertins, W. Süsshuth und R. Jochen. *Modellierungsmethoden für rechnerintegrierte Produktionsprozesse*. C. Hanser Verlag, München-Wien, 1994.

[Pau93] G. Paul. Produkt(daten)modellierung als Grundlage der Integration von CAx– und PPS–Komponenten. Preprint ITI 1/93, Fakultät für Informatik, Universität Magdeburg, Mai 1993.

[Pis93] P. Pistor. Objektorientierung in SQL3: Stand und Entwicklungstendenzen. *Informatik-Spektrum*, 16(2), 1993.

[Pu87] C. Pu. Superdatabases: Transactions Across Database Boundaries. *Bulletin of the IEEE Technical Committee on Data Engineering*, 10(3):19–25, 1987.

[PWPH94] G. Paul, H. Wierschin, R. Paul und M. Hofmann. Produktdatenverwaltungssystem - Methode und Werkzeug zur Unternehmensintegration. Bericht 94-02, ITI, Universität Magdeburg, 1994.

[PWPH95] G. Paul, H. Wierschin, R. Paul und M. Hofmann. Produktdatenverwaltungssysteme – Methode und Werkzeug zur Unternehmensintegration. *CIM Management*, S. 49–54, April 1995.

[Rad94] E. Radeke. Efendi: Federated Database System of Cadlab. Technical Report 39/94, University of Paderborn & Siemens Nixdorf, Oktober 1994.

[Rah94] E. Rahm. *Mehrrechner-Datenbanksysteme.* Addison-Wesley, Bonn, 1994.

[RB95] M. Rüger und T. Behlau. Create!: an Object-Oriented IDE for Discrete Event Simulation. In C. Alexopoulos, K. King, W. R. Lilegelon und D. Goldsman, Herausgeber, *Proc. of the 1995 Winter Simulation Conf. WSC'95, Arlington, USA*, S. 775–780, Dezember 1995.

[RS94] E. Radeke und M. H. Scholl. Federation and Stepwise Reduction of Database Systems. In W. Litwin und T. Risch, Herausgeber, *Proc. of the 1st Int. Conf. on Applications of Databases (ADB'94), Vadstena, Sweden*, S. 381–399. LNCS 819, Springer-Verlag, Juni 1994.

[RS95] E. Radeke und M. H. Scholl. Functionality for Object Migration among Distributed, Heterogenous, Autonomous Databases Systems. In O. Bukhres, T. Özsu und M.-C. Shan, Herausgeber, *Proc. of the 5th Int. Workshop on Research Issues in Data Engineering: Distributed Object Management (RIDE-DOM'95), Taipei, Taiwan*, S. 58–66. IEEE Computer Society Press, März 1995.

[Rüg95] M. Rüger. Building Discrete Event Simulators with Create! In F. Breitenecker und I. Husinsky, Herausgeber, *Proc. of the 1995 European Simulation Conf. EUROSIM95, Vienna, Austria.* Elsevier, Amsterdam, September 1995.

[Saa93] G. Saake. *Objektorientierte Spezifikation von Informationssystemen.* Teubner, Stuttgart/Leipzig, 1993.

[SCG91] F. Saltor, M. Castellanos und M. Garcia-Solaco. Suitability of Data Models as Canonical Models for Federated Databases. In J. Clifford und R. King, Herausgeber, *Proc. of the 1991 ACM SIGMOD Int. Conf. on Management of Data, Denver, Colorado, SIGMOD RECORD 20(2)*, S. 44–48. ACM Press, Juni 1991.

[SCG93] F. Saltor, Castellanos und M. Garcia-Solaco. Overcoming Schematic Discreprancies in Interoperable Databases. In D. K. Hsiao, E. J. Neuhold und R. Sacks-Davis, Herausgeber, *Interoperable Database Systems (DS-5), Proc. of the IFIP WG 2.6 Database Semantics Conf., Lorne, Victoria, Australia, November, 1992*, S. 191–205. North-Holland, 1993.

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

[SK93] A. Sheth und V. Kashyap. So Far (Schematically) yet So Near (Semantically). In D. K. Hsiao, E. J. Neuhold und R. Sacks-Davis, Herausgeber, *Interoperable Database Systems (DS-5), Proc. of the IFIP WG 2.6 Database Semantics Conf., Lorne, Victoria, Australia, November, 1992*, S. 283–312. North-Holland, 1993.

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

[SP91] S. Spaccapietra und C. Parent. Conflicts and Correspondence Assertions in Interoperable Databases. *ACM SIGMOD RECORD*, 20(4), Dezember 1991.

[SPD92] S. Spaccapietra, C. Parent und Y. Dupont. Model Independent Assertions for Integration of Heterogeneous Schemas. *The VLDB Journal*, 1(1):81–126, 1992.

[Spe90] X/Open Preliminary Specification. *Distributed Transaction Processing: The XA Interface.* X/Open Ltd., Reading, Berkshire, England, April 1990.

[Spe92] X/Open Specification. *Distributed Transaction Processing: The TxRPC Specification.* X/Open Ltd., Reading, Berkshire, England, 1992.

[SS95] I. Schmitt und G. Saake. Managing Object Identity in Federated Database Systems. In M. Papazoglou, Herausgeber, *Proc. of the 14th Int. Conf. on Object-Oriented and Entity-Relationship Modeling (OOER'95), Gold Coast, Australia*, S. 400–411. LNCS 1021, Springer-Verlag, Dezember 1995.

[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, *Proc. of the 5th Int. Workshop on Research Issues on Data Engineering: Distributed Object Management (RIDE-DOM'95), Taipei, Taiwan*, S. 108–115. IEEE Computer Society Press, März 1995.

[Str93] Jürgen Strauss. Eine kurze Einführung in Concurrent Engineering. Cadlab Report 1/93, Cadlab Paderborn, Germany, 1993.

[Sun93] R. L. Sunday. *Information Technology - Open Systems Interconnection - Remote Database Access - Part 2: SQL Spezialization.* ISO/IEC JTC 1/ SC21/ WG 3. DOC. Nr. ISO/IEC 9579-2:1993(E), Februar 1993.

[SW91] H.-J. Schek und G. Weikum. Erweiterbarkeit, Kooperation, Föderation von Datenbanksystemen. In H.-J. Appelrath, Herausgeber, *Proc. GI-Fachtagung "Datenbanksysteme in Büro, Technik und Wissenschaft", Kaiserslautern*, S. 38–71. Informatik-Fachberichte, Springer-Verlag, März 1991.

[TC193] ISO TC184/SC4. *Product Data Representation and Exchange – Part 1: Overview and Fundamental Principles.* ISO/DIS 10303-1, 1993.

[TC95] C. Türker und S. Conrad. Aktive Mechanismen zur Konsistenzsicherung in föderierten Datenbanksystemen. In C. Eckert, H.-J. Klein und T. Polle, Herausgeber, *7. GI-Workshop "Grundlagen von Datenbanken", Bad Salzdetfurth*, S. 148–152. Hildesheimer Informatik Berichte, Juni 1995.

[Wei95] F. Weiler. Ein generativer Ansatz zur Realisierung eines objektorientierten Zugriffs auf relationale Datenbstände im Rahmen der Integration heterogener DBMS. Diplomarbeit, Universität Kaiserslautern, Januar 1995.

[Zie95] D. Ziems. Kooperation und Integration - Aktuelle Herausforderungen an Planungsprozesse. In *Planung von Logistiksystemen - Kooperation und Integration 95, Magdeburg*, S. 3–22, Oktober 1995.

[ZWB95] D. Ziems, O. Wolter und F. Barth. Adaptive Strukturbestimmung im integrierten und durchgängigen Gestaltungsprozeß. In *Fachtagung "Integration von Bild, Modell und Text", Magdeburg*, S. 207–218, März 1995.