Sigma$_{Bench}$ – Ein Werkzeug zum Entwurf föderierter Datenbanken *

Ingo Schmitt, Andreas Ebert, Michael Höding, Can Türker,

Institut für Technische Informationssysteme
Otto-von-Guericke-Universität Magdeburg
Postfach 4120, D–39016 Magdeburg
E-mail: {schmitt|aebert|hoeding|tuerker}@iti.cs.uni-magdeburg.de

Zusammenfassung

In diesem Papier geben wir einen allgemeinen Überblick über das Projekt **SIGMA**$_{Bench}$ welches das Ziel hat, ein Werkzeug für den systematischen Entwurf föderierter Datenbanken zu entwickeln. Insbesondere soll die Extraktion von Schemata aus Dateien, die Schematransformation und -integration, sowie die Behandlung von Integritätsbedingungen unterstützt werden.

1 Einführung

In vielen Unternehmen existieren heterogene Informationssysteme, die ihre Daten unterschiedlich verwalten. Einige von diesen Systemen basieren auf Datenbankmanagementsystemen, andere legen ihre Daten direkt in Form von Dateien ab. In den verschiedenen Informationssystemen werden oft Daten über denselben Weltausschnitt verwaltet. Diese Redundanz birgt das Potential für Konsistenzprobleme im Sinne von Aktualität in sich.

Weiterhin benötigen neue Anwendungen oft eine einheitliche Datenbankschnittstelle zu den verteilt verwalteten Daten. Um diese beiden angesprochenen Probleme zu lösen ist es in der Praxis oft nicht möglich, diese Daten in eine zentrale Datenbank zu migrieren, da die lokalen Informationssysteme weiterhin unverändert funktionieren sollen. Einen Ausweg aus dem Dilemma des Zwangs zur Integration und der Wahrung von lokaler Autonomie bestehender Datenverwaltungssysteme bieten föderierte Datenbanksysteme [SL90, Con97]. Diese Systeme führen eine Integration nicht auf der Datenebene, sondern "virtuell" auf der Schemaebene durch. Das bedeutet, daß die Daten weiterhin von den lokalen Datenverwaltungssystemen verwaltet werden und für globale Anwendungen ein integriertes Schema einer globalen Datenbank simuliert wird.

Bevor man föderierte Datenbanksysteme einsetzen kann, muß ein Entwurf der föderierter Datenbank durchgeführt werden. Dieser sehr komplexe Vorgang umfaßt das Extrahieren von Schemata aus den lokalen Datenverwaltungssystemen, die Schemaintegration, welche in erster Linie die Beseitigung von Datenmodell- und Schemaheterogenität bedeutet, und das Ableiten externer globaler Sichten. Das Grundproblem beim Entwurf föderierter Datenbanken ist der erforderliche Umgang mit der Semantik der dezentral gespeicherten Daten. Ohne Betrachtung dieser Semantik ist die Erzeugung einer virtuellen, homogenen und globalen Datenbankschnittstelle nicht möglich. Oft liegt diese Semantik in den lokalen Systemen nicht explizit oder nur unvollständig vor. Daher muß in den Entwurfsprozeß menschliches Hintergrundwissen einbezogen werden. Die Frage lautet nun, inwieweit kann ein Entwurfswerkzeug den Entwurf unterstützen? Zum einen kann innerhalb eines solchen Werkzeuges eine Entwurfsmethode implementiert sein, die dem Designer hilft, die richtigen Schritte zur richtigen Zeit durchzuführen. Weiterhin kann ein solches System die Ausgangsdaten, Zwischenergebnisse und Entwurfsergebnisse in einer nutzerfreundlichen Form präsentieren. Insbesondere ermöglicht aber ein Entwurfswerkzeug die Anwendung von Algorithmen, die die einzelnen Schritte des Entwurf unterstützen. Im geplanten Entwurfswerkzeug **SIGMA**$_{Bench}$ sollen entwurfsunterstützende Algorithmen nutzbringend umgesetzt und damit validiert werden.

*Diese Arbeit wird teilweise gefördert vom Land Sachsen-Anhalt unter FKZ: 1987A/0025 "Föderierung heterogener Datenbanksysteme und lokaler Datenhaltungskomponenten zur systemübergreifenden Integritätssicherung" – "**SIGMA**$_{FDB}$ (**S**_chema_ **I**_ntegration and_ **G**_lobal integrity_ **M**_aintenance_ **A**_pproach for_ **F**_ederated_ **D**_ata_ **B**_ases_)"

Im nächsten Abschnitt wollen wir die wesentlichen Ziele des Entwurfswerkzeuges **SIGMA**$_{Bench}$ skizzieren. Danach gehen wir im Abschnitt 3 auf die Konzeption dieses Werkzeuges ein. Die technischen Aspekte werden im Abschnitt 4 beschrieben. Eine kurze Zusammenfassung beschließt diesen Beitrag.

2 Zielbeschreibung

Eines der Ziele des **SIGMA**$_{FDB}$-Projektes [HTJ$^+$95] ist die Erforschung des Entwurfs föderierter Datenbanken. Die schon vorhandenen Forschungsergebnisse umfassen sowohl Vorschläge des methodischen Vorgehens als auch direkt umsetzbare Algorithmen. Innerhalb des Projektes **SIGMA**$_{Bench}$ werden durch die Implementierung eines Entwurfsprototyps in erster Linie folgende Ziele erfolgt:

- **Validierung der Forschungsergebnisse:** Durch die Umsetzung der Forschungsergebnisse soll ihre Umsetzbarkeit nachgewiesen werden. Das Entwurfswerkzeug kann anhand von Beispielszenarien getestet werden. Die Testergebnisse sind besonders für die Ausrichtung weiterer Forschung wichtig.

- **Präsentation:** Mit Hilfe des Prototyps **SIGMA**$_{Bench}$ können die Forschungsergebnisse des **SIGMA**$_{FDB}$-Projektes anwendungsnah präsentiert werden. Somit sind sie für potentiellen Anwender und andere Forschungsgruppen besser nachvollziehbar, als dies durch ihre reine Veröffentlichung möglich ist.

Wie bereits erwähnt, kann der Entwurf nicht vollständig automatisiert werden, da Interaktionen mit Experten, die das Hintergrundwissen mitbringen, nötig sind. Hier soll der Prototyp **SIGMA**$_{Bench}$ den Entwurfsprozeß soweit wie möglich unterstützen. In der Regel gibt es nicht einen einzelnen Weg, eine föderierte Datenbank zu entwerfen, sondern eine Vielzahl von verschiedenen Wegen. Durch die geeignete Präsentation verschiedender Varianten soll dem Entwerfer bei der Entscheidung über alternative Entwurfsschritte geholfen werden. Konkret soll das Werkzeug folgende Schritte des Datenbankentwurfs unterstützen:

- **Schemabereitstellung:** In eine Föderierung können neben Datenbanken auch Dateien einbezogen werden. Dies ist notwendig, da viele Informationssysteme wichtige Daten nicht in Datenbanken, sondern in selbst verwalteten Dateien ablegen. Von diesen Dateien ist in der Regel nicht bekannt, wie sie strukturiert sind, das heißt, das zugrundeliegende Schema ist unbekannt. Oft sind in solchen Dateien Metadaten mit den eigentlichen Daten vermischt. Ziel dieses Entwurfschrittes ist es, mit geeigneten "data mining"-Verfahren ein passendes Schema für solche Dateien zu extrahieren und zu validieren. Mit Hilfe dieses Schemas und weiterer Strukturierungsinformationen soll es möglich sein, eine datenbankähnliche Schnittstelle zu solchen Dateien zu erstellen. Da alle extrahierten Schemata die Grundlage für die Schemaintegration sein sollen, liegen sie in einem einheitlichen Datenmodell vor.

  Ein anderer Punkt der Schemabereitstellung ist das Extrahieren von konzeptionellen Schemata aus vorhandenen Datenbanksystemen. Dafür ist ein Zugriff auf Katalogdaten notwendig.

  Als letzter Punkt dieses Schrittes soll es natürlich auch möglich sein, Schemata interaktiv einzugeben.

- **Schematransformation und -integration:** Nachdem die bereitgestellten Schemata in ein gemeinsames Datenmodell transformiert wurde, muß bei der Schemaintegration die Heterogenität auf der Schemaebene überwunden werden. Dazu müssen in erster Linie Konflikte erkannt werden. Es ist wichtig, daß das Entwurfssystem auf einer geeigneten Konfliktklassifikation aufbaut. Die Menge der unterschiedlichen Konfliktarten soll klein aber möglichst vollständig sein. Die Komplexität der Schemaintegration kann weiter minimiert werden, wenn die Konfliktarten sich nicht gegenseitig beeinflussen. Ein zusätzlicher Aspekt ist die Reihenfolge der Erkennung und Auflösung der verschiedenen Konfliktarten. Eine geschickt gewählte Reihenfolge kann den Entwurfsprozeß erheblich vereinfachen. Neben der Erzeugung eines integrierten Schemas muß auch die Spezifikation, wie auf globale Operationen reagiert werden soll, durchgeführt werden. Dies ist besonders wichtig für globale Änderungsoperationen wie Ändern von Attributwerten, Einfügen und Entfernen von globalen Objekten. Zum Beispiel muß in manchen Fällen explizit angegeben werden, in welchen lokalen Datenverwaltungssystemen neu erzeugte globale Objekte abgelegt werden sollen. Bei Änderungsoperationen muß grundsätzlich die Integrität der Datenbestände gewährleistet werden.

- **Anreicherung globaler Schemata um neue, globale Integritätsbedingungen:** Oftmals stellt man nach der Schemaintegration fest, daß gewisse Integritätsbedingungen auf der Ebene des globalen Schemas notwendig sind, etwa um eine konsistente Artikelbeschreibung bzw. -numerierung innerhalb eines großen Unternehmens zu garantieren. Weitere Beispiele für solche komponentendatenbankübergreifende Integritätsbedingungen sind globale Schlüsselbedingungen (siehe dazu auch [SS95]), referentielle Integritätsbedingungen sowie komplexe Bedingungen formuliert mit Hilfe der bekannten Aggregatfunktionen **min**, **max**, **sum**, **avg** und **count**. Solche globalen Integritätsbedingungen müssen bereits beim Entwurf der föderierten Datenbank berücksichtigt und geeignete Vorkehrungen zur Überwachung von potentiell integritätsverletzenden Operationen getroffen werden.

- **Sichtenableitung:** Nach der Schemaintegration liegen ein integriertes Schema sowie Transformationsinformationen zu lokalen Schemata vor. Da ein föderiertes Datenbanksystem genau wie jedes andere Datenbanksystem logische Datenunabhängigkeit unterstützen soll, muß es möglich sein, verschiedene anwendungsspezifische Schemata auf der Basis des integrierten Schemas zu erzeugen. Diese Sichtableitung kann als ein inverser Schritt zur Schemaintegration gesehen werden. Die Schemaintegration soll die Heterogenität überwinden können, die bei der Sichtableitung generiert werden kann. Als Sonderfall für föderierte Datenbanksysteme sollen externe globale Schemata ableitbar sein, die äquivalent oder ähnlich zu lokalen Schemata sind. Dies ist wichtig, wenn man lokale Anwendungen auf die globale Schnittstelle migrieren will.

- **Generierung von Abbildungsdaten:** Als Ergebnis des Entwurfsprozesses entstehen neue Schemata sowie Informationen über Schema- und Datentransformationen. Diese Informationen sollen in diesem Schritt so aufbereitet werden, daß sie direkt als Eingabe für ein föderiertes Datenbanksystem dienen können.

Die hier beschriebenen Ziele sind sehr weitreichend. Sie beschreiben die Funktionalität des angestrebten Entwurfswerkzeuges. Um diese Funktionalität zu erreichen, muß sowohl weitere Forschungsarbeit als auch Implementierungsarbeit geleistet werden. Nichtsdestotrotz baut dieses Werkzeug auf einigen schon vorhandenen Forschungsergebnissen auf, die wir kurz im nächsten Abschnitt beschreiben wollen.

3 Grobkonzeption

In diesem Abschnitt soll gezeigt werden, wie wir den Entwurfsprozeß unterstützen wollen. Dabei gehen wir nicht auf organisatorische und implementierungstechnische Aspekte ein. Diese Aspekte werden im nächsten Abschnitt erörtert.

Die Abbildung 1 zeigt den groben Ablauf des Entwurfsprozesses. Zum einen umfaßt der Entwurfsprozeß Aufgaben der Dateianalyse, -modellierung und Schemavalidierung und zum anderen den Prozeß der Schemaintegration und Ableitung externer Schemata. Dabei wird auf verschiedenen Schemaebenen in unterschiedlichen Datenmodellen gearbeitet. Im Rahmen des Projektes **SIGMA**$_{FDB}$ existieren folgende

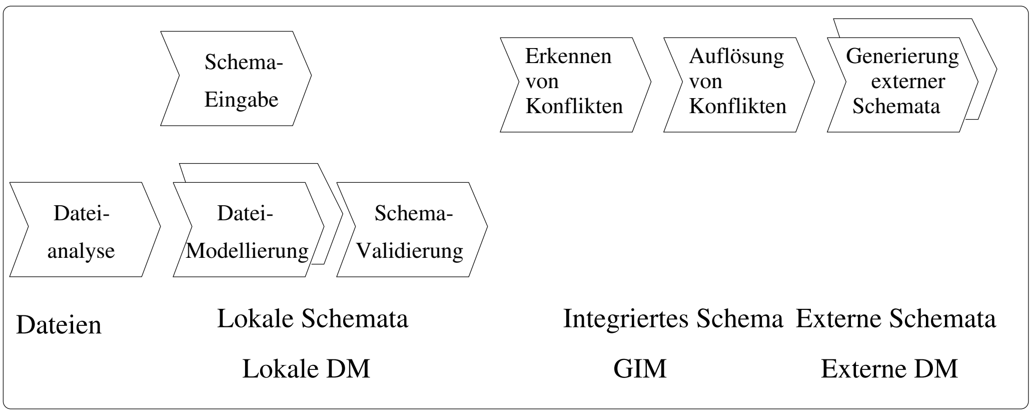

Abbildung 1: Entwurfsprozeß

Forschungsergebnisse, die innerhalb des Entwurfswerkzeuges umgesetzt werden sollen:

- **Schemagenerierung für Dateienverbünde**[1]**:** Zur Integration von Dateienverbünden in Föderierte Datenbanksysteme ist es notwendig, ein konzeptionelles Schema zu generieren, das die (mögliche) Struktur des Dateienverbundes adäquat, vollständig und anschaulich beschreibt. Ausgangsinformationen für diesen Schritt sind

  - *Beispieldateien*, die eine mögliche Ausprägung des Dateienverbundes repräsentieren,
  - *Zugriffsinformation*, die durch Überwachung des Zugriffs von Applikationsprogrammen auf das Dateisystem gewonnnen werden,
  - *Quelltexte* von Applikationsprogrammen, in denen das Schema (oder Teile davon) kodiert sind, und
  - *Hintergrundwissen* aus Dokumentationen, Beispielsitzungen oder als Wissen über das Anwendungsszenario.

  Im allgemeinen ist es nicht möglich, auf Basis der ersten drei Informationsquellen ein konzeptionelles Schema abzuleiten. Der Schemaableitungsprozeß ist deshalb immer im Sinne eines Remodellierungsprozesses zu gewissen Teilen manuell durchzuführen, das heißt mit Hilfe des Hintergrundwissen muß ein Schema modelliert werden. Um jedoch die Qualität und Effizienz dieses Entwurfsschrittes zu erhöhen, sollte dieser Prozeß so weit wie möglich automatisch unterstützt werden.

  Wir gehen davon aus, daß sich vorhandene Dateiverbünde klassifizieren lassen und daß für jede Klasse spezielle Methoden und Werkzeuge notwendig sind, um die Schemaableitung zu unterstützen. Ein Vorschlag für eine empirische Klassifikation wurde in [Höd96b] vorgestellt.

  Aufbauend auf diesen Annahmen schlagen wir ein systematisches Vorgehen vor [Höd96a]. Der erste Schritt dabei ist die Anreicherung eines vorhandenen allgemeinen Wörterbuches um anwendungsspezifische Daten und Metadaten zur (teilweisen) Formalisierung des Hintergrundwissens. Dieses erweiterte Wörterbuch wird dann unter Berücksichtigung vorhandener Zugriffsinformationen auf die Beispieldateien angewandt und somit eine automatische Zuordnung des Dateienverbundes zu einer Klasse der Klassifikation durchgeführt. Auch im nächsten Schritt kommt wieder das Wörterbuch zum Einsatz. Mittels klassenspezifischer Methoden entsprechend der Beispieldateien werden Vorschläge generiert, die den Entwerfer bei der Modellierung eines adäquaten Schemas unterstützen. Das modellierte Schema kann dann anhand der Beispieldateien oder, sehr aufwendig, durch Generierung künstlicher Dateien und probeweises Einlesen durch Applikationen validiert werden. Zur Modellierung der Strukturbeschreibung kann man die Sprache SGML [ISO86] nutzen. Im nächsten Schritt sollte, ebenfalls automatisch unterstützt, die Generierung eines verbesserten Schemas in einem bekannten Datenmodell erfolgen. Dabei kann man sich das Vorhandensein einer Ausprägung der Datenbank, die in Form der Beispieldateien zur Verfügung steht, zunutze machen.

- **Schemaintegration und -ableitung:** Innerhalb dieses Schrittes muß die Heterogenität der zu integrierenden Schemata überwunden werden. Die Datenmodellheterogenität wird durch das Transformieren der zu integrierenden Schemata in ein einheitliches Datenmodell beseitigt. Wir verwenden als gemeinsames Datenmodell das eigens entwickelte Datenmodell GIM [Sch95]. Dieses Datenmodell ist ein semantisch armes Datenmodell, welches durch seine einfachen Konzepte das Anwenden von entwurfsunterstützenden Algorithmen gestattet. Die Transformation der zu integrierenden Schemata ist mit einem Normalisierungsprozeß verbunden, da relativ wenig Modellierungskonzepte in GIM unterstützt werden. Allein dadurch wird neben Datenmodellheterogenität auch Schemaheterogenität überwunden.

  Nachdem die Eingabeschemata in GIM vorliegen, müssen Schemakonflikte sowohl erkannt als auch aufgelöst werden. In [Sch96] wird eine Konfliktklassifikation durchgeführt. Für die einzelnen Konfliktarten wird dort eine Reihenfolge vorgeschlagen, in der die einzelnen Konflikte erkannt und aufgelöst werden sollten.

  Eine besondere Rolle spielt der extensionale und der intensionale Konflikt innerhalb der Schemaintegration. Mit dem extensionalen Konflikt ist gemeint, daß Informationen über dasselbe "Realwelt"-Objekt in unterschiedlichen Klassen gespeichert sein können. Der intensionale Konflikt drückt aus, daß über dasselbe Objekt in verschiedenen Klassen sowohl unterschiedliche als auch gleiche Informationen in Form von Attributwerten verwaltet werden können. Zur Auflösung dieser beiden

[1] Unter einem *Dateienverbund* verstehen wir eine Menge semantisch zusammengehöriger Dateien, wie z.B Dateien die durch eine Applikation zur persistenten Datenhaltung genutzt werden oder HTML-Dateien eines WWW-Sites

Konfliktarten schlagen wir eine extensionale und eine intensionale Zerlegung [SS96a] vor. Als Ergebnis erhalten wird das integrierte GIM-Schema. Da GIM nur als Datenmodell für den föderierten Datenbankentwurf entwickelt wurde, kann das integrierte Schema nicht für globale Anwendungen genutzt werden. Statt dessen bildet es die Grundlage für die Ableitung externer Schemata. Dazu sind Algorithmen zum Zusammensetzen von GIM-Klassen zu Klassen der externen Schemata anwendbar. Es werden Algorithmen aus dem Bereich der formalen Begriffsanalyse [GW96] an die Besonderheiten der Schemaintegration angepaßt. Auf der Grundlage dieser und im Rahmen des Projektes **SIGMA**$_{FDB}$ entwickelter Algorithmen [SS96c, SS96b] können verschiedene externe Sichten abgeleitet werden.

- **Maßnahmen zur komponentensystemübergreifenden Integritätssicherung:** Zur Wahrung der globalen Integrität einer föderierten Datenbank müssen die Operationen, die zu einer Integritätsverletzung führen können, kontrolliert werden. Dazu müssen zunächst Integritätsbedingungen aufgestellt werden, die die Korrektheit einer föderierten Datenbank beschreiben. Beim Transformations- und Integrationsprozeß müssen in den Komponentendatenbanken existierende Integritätsbedingungen berücksichtigt und neue Integritätsbedingungen definiert werden [CHJ+96, CST96].

  In [TC95, CT95, TC96, Tür96, CT97] haben wir eine FDBS-Architektur vorgestellt, die darauf ausgerichtet ist, eine Basis für die Integration von integritätserhaltenden bzw. -wiederherstellenden Mechanismen zu bieten. Die Grundidee bei dieser Architektur ist es, das ECA-Regelkonzept zur Überwachung lokaler und globaler Integritätsbedingungen einzusetzen. Hierfür sind im Entwurfsprozeß einer föderierten Datenbank die lokalen und globalen Operationen zu ermitteln, die eine Integritätsverletzung nach sich ziehen können. Dies kann mit Hilfe der Metadaten der Datenbankföderation geschehen. Für die Überwachung all dieser Operationen sind geeignete ECA-Regeln, soweit es geht, automatisch zu generieren.

  In diesem Zusammenhang müssen verschiedene Probleme beachtet werden:

  – Zusammenspiel von existierenden und neuen integritätserhaltenden Maßnahmen: Oftmals erfolgt die Integritätssicherung (wenn überhaupt) in Anwendungsprogrammen außerhalb der Kontrolle des Datenbankmanagementsystems. Solche Aspekte müssen ebenfalls bei der Generierung von integritätserhaltenden Regeln in Betracht gezogen werden.

  – Konfluenz zwischen lokalen und globalen Integritätsregeln: Das aus den zentralen Datenbanksystemen bekannte Problem der Reihenfolgeabhängigkeit von gleichzeitig getriggerten Regeln wird in föderierten Datenbankumgebungen weiter verschärft. Ferner sind auch die Nebeneffekte von integritätswiederherstellenden Aktionen zu berücksichtigen. Da ECA-Regeln weitere ECA-Regeln triggern können, kann der Fall eintreten, daß der Effekt der Integritätswiederherstellung nicht erzielt wird, weil sich die getriggerten Regeln gegenseitig beeinflusssen. Dieser Aspekt muß soweit es geht durch das Entwurfstool schon beim Entwurf verhindert werden.

4 Technische Aspekte

In diesem Abschnitt soll auf einige organisatorische und implementierungsspezifische Aspekte eingegangen werden. Das Entwurfswerkzeug **SIGMA**$_{Bench}$ wird im Rahmen des Projektes **SIGMA**$_{FDB}$ von den Projektbeteiligten als auch von studentischen Hilfskräften entwickelt. Insgesamt ist nur eine Prototypentwicklung geplant.

Ein wichtige Implementierungsentscheidung war die Entscheidung für die Programmierung der Nutzeroberfläche in Java. Damit kann die Oberfläche dieses Entwurfswerkzeugs über das Internet verfügbar gemacht werden. Außerdem ermöglicht die Verwendung von Java eine hohe Plattformunabhängigkeit. Abbildung 2 zeigt einen Teil der vorläufigen Oberflächengestaltung. Der in Abbildung 2 dargestellte Teil der Nutzeroberfläche dient zum Editieren der Schemata und zur Eingabe von extensionalen Überlappungen zweier Schemata. Links oben wird die Struktur des Repositories angezeigt. Rechts daneben können ausgewählte Schemata editiert werden. Im unteren Teil des dargestellten Teils der Oberfläche können für zu integrierende Schemata extensionale Überlappungen angegeben werden, die für die spätere Schemaintegration benötigt werden. Weitere Teile der Oberfläche stellen Funktionen zum Aufruf weiterer Programmodule bereit.

Eine weitere wichtige Implementierungsentscheidung betraf die Nutzung eines einheitlichen Repositories, das in einer relationalen Datenbank abgelegt wird. Die Verwendung einer Datenbank als Repository

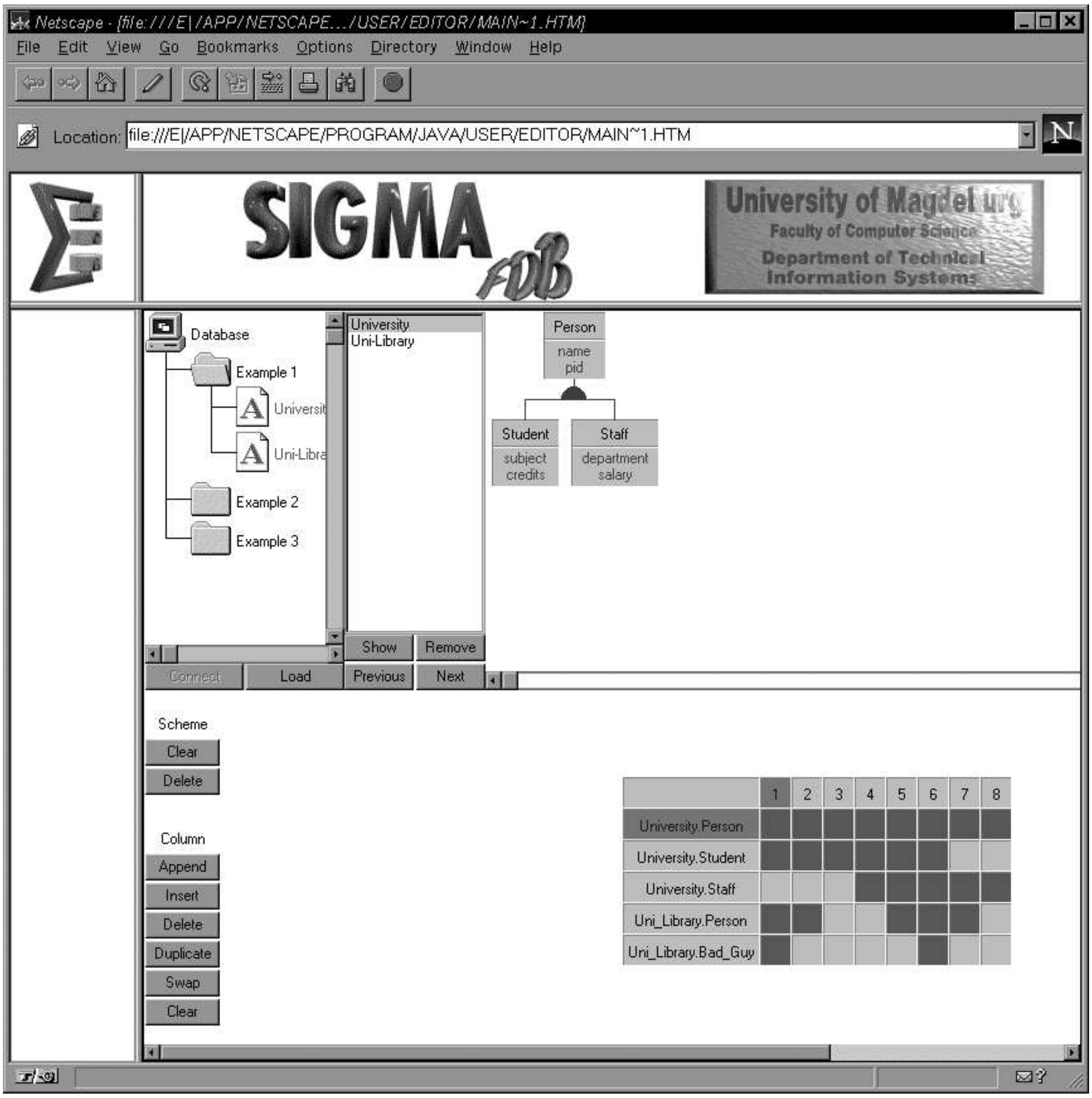

Abbildung 2: Java Schnittstelle

ermöglicht den Zugriff von verschiedenen, separat arbeitenden Tools, die die benötigten Daten aus dieser Datenbank einlesen und die Resultate dort wieder ablegen. Die grobe Softwarearchitektur wird in Abbildung 3 gezeigt. Die einzelnen Bausteine kommunizieren über die Datenbank. Dadurch kann das gesamte System als ein offener Baukasten gesehen werden, bei dem aufgrund der fest definierten Datenbankschnittstelle relativ leicht verschiedene Module ausgetauscht beziehungsweise ergänzt werden können. Der Zugriff auf die Datenbank von der Java-Oberfläche erfolgt entweder über JDBC oder mittels einer CORBA-Realisierung.

Die Abbildung 3 soll den modularen Charakter der **SIGMA**$_{Bench}$ betonen. Die einzelnen Module entsprechen den schon in vorangegangenen Abschnitten beschriebenen geplanten Funktionsgruppen.

Die jeweiligen Module sind in C++ implementiert. Die Implementierung in C++ hat den Vorteil, daß später eine leichte Portierung nach Java erfolgen kann. Es wird bei der Implementierung unter C++ darauf geachtet, daß eine Unabhängigkeit vom Compiler erhalten bleibt.

Die Entwicklung des Projekts erfolgt in mehreren Etappen. Zuerst wurden wesentliche Konflikte bei der Schemaintegration und Schemaableitung behandelt. In der darauffolgenden Etappe werden Integritätsbedingungen bzw. -regeln bei der Integration berücksichtigt. Parallel zur Implementierung weiterer Ableitungsalgorithmen wird an der Analyse von Dateien gearbeitet.

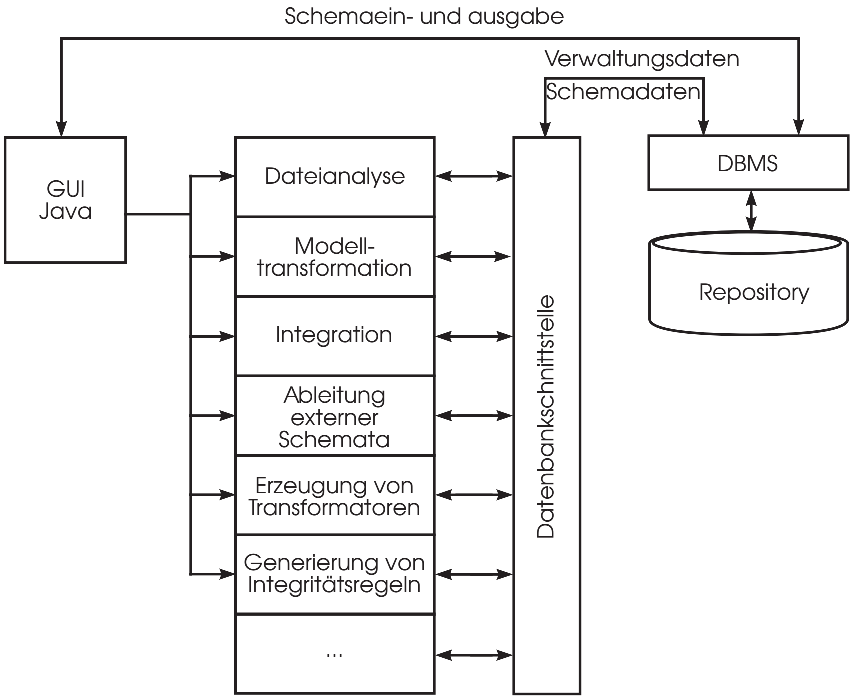

Abbildung 3: Struktur von **SIGMA**$_{Bench}$

5 Zusammenfassung

Wir haben in diesem Beitrag versucht, die Ziele und Ideen des Projektes **SIGMA**$_{Bench}$ vorzustellen. Innerhalb dieses Projektes sollen vorhandene Forschungsergebnisse in Form eines interaktiven Entwurfswerkzeuges implementiert werden. Da die Forschung im Rahmen des Projektes **SIGMA**$_{FDB}$ zum Entwurf föderierter Datenbanken noch nicht abgeschlossen ist, befindet sich dieses Entwurfswerkzeug noch in der Entwicklungsphase. Deshalb haben wir versucht, nicht zu detailliert auf schon implementierte Aspekte einzugehen, sondern eher das Gesamtanliegen des Projektes **SIGMA**$_{Bench}$ zu beschreiben.

Dieser Bericht kann weiterhin dazu verwendet werden, einen allgemeinen Überblick über die Forschungsbemühungen im Rahmen des Projektes **SIGMA**$_{FDB}$ zu bekommen. Dazu wird besonders im Abschnitt 3 auf wesentliche Forschungsergebnisse mit den dazugehörigen Literaturverweisen eingegangen.

Literatur

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

[Con97] S. Conrad. Föderierte Datenbanken: Konzepte der Datenintegration. Habilitationsschrift, Otto-von-Guericke-Universität Magdeburg, Fakultät für Informatik, 1997. To appear.

[CST96] S. Conrad, I. Schmitt, and C. Türker. Transformation von Integritätsbedingungen bei der Schemaintegration (Extended Abstract). In W. Hasselbring, editor, *Kurzfassungen des 2. Workshops "Föderierte Datenbanken", Dortmund, 12.-13. Dezember 1996*, 1996.

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

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

[GW96] B. Ganter and R. Wille. *Formale Begriffsanalyse.* Springer-Verlag, Berlin, Heidelberg, 1996.

[Höd96a] M. Höding. An Approach for Integration of File Based Systems into Database Federations. In *Proc. of the Int. Workshop on Heterogeneous Information Management, Prague, Czech Republic, November 1996*, 1996. *To appear.*

[Höd96b] M. Höding. Föderierung von Dateien: Finden, Verwalten und Nutzen von Metainformationen. In S. Conrad, G. Saake, I. Schmitt, and C. Türker, editors, *Kurzfassungen – 8. Workshop "Grundlagen von Datenbanken", Friedrichsbrunn/Harz (28.05.–31.05.96)*, pages 46–50. Institut für Technische Informationssysteme, Universität Magdeburg, May 1996.

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

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

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

[Sch96] I. Schmitt. Operations of Homogenizing and Heterogenizing in Federated Database Systems. Preprint Nr. 5, Fakultät für Informatik, Universität Magdeburg, September 1996.

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

[SS95] I. Schmitt and G. Saake. Managing Object Identity in Federated Database Systems. In M. Papazoglou, editor, *OOER'95: Object-Oriented and Entity-Relationship Modeling, Proc. of the 14th Int. Conf., Gold Coast, Australia*, pages 400–411. LNCS 1021, Springer-Verlag, Berlin, December 1995.

[SS96a] I. Schmitt and G. Saake. Flexible Generation of Global Integrated Schemata using GIM. In S. Conrad, M. Höding, S. Janssen, and G. Saake, editors, *Kurzfassungen des Workshops "Föderierte Datenbanken", Magdeburg, 22.04–23.04.96*, pages 29–43. Bericht 96–01, Institut für Technische Informationssysteme, Universität Magdeburg, April 1996.

[SS96b] I. Schmitt and G. Saake. Integration of Inheritance Trees as Part of View Generation for Database Federations. In B. Thalheim, editor, *Conceptual Modelling — ER'96, Proc. of the 15th Int. Conf., Cottbus, Germany, October 1996*, pages 195–210. LNCS 1157, Springer-Verlag, Berlin, 1996.

[SS96c] I. Schmitt and G. Saake. Schema Integration and View Generation by Resolving Intensional and Extensional Overlappings. In K. Yetongnon and S. Hariri, editors, *Proc. of the 9th ISCA Int. Conf. on Parallel and Distributed Computing Systems (PDCS'96), Dijon, France*, pages 751–758, September 1996.

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

[TC96] C. Türker and S. Conrad. Using Active Mechanisms for Global Integrity Maintenance in Federated Database Systems. In S. Conrad, M. Höding, S. Janssen, and G. Saake, editors, *Kurzfassungen des Workshops "Föderierte Datenbanken", Magdeburg, 22.04–23.04.96*, pages 51–65. Bericht 96–01, Institut für Technische Informationssysteme, Universität Magdeburg, April 1996.

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