SIGMA$_{FDB}$: Overview of the Magdeburg-Approach to Database Federations*

M. Höding, K. Schwarz, S. Conrad, G. Saake, S. Balko, A. Diekmann,
E. Hildebrandt, K.-U. Sattler, I. Schmitt, and C. Türker

Otto-von-Guericke-Universität Magdeburg
Institut für Technische und Betriebliche Informationssysteme
Postfach 4120, D–39016 Magdeburg, Germany
sigmafdb@iti.cs.uni-magdeburg.de

Abstract. The **SIGMA**$_{FDB}$ project attempts to offer an approach to schema integration and integegrity constraint maintenance in the field of federated database systems. In this extended abstract, we present our view on federated database systems and sketch the main results of our group's work. Especially, we briefly discuss different research aspects and implementation activities.

1 Introduction

Our research activities in the field of federated database systems (FDBS) are closely related to the **SIGMA**$_{FDB}$ project[1] which is funded by the german state Sachsen-Anhalt since 1995. The different research aspects and implementation activities of our group are based on the following three application scenarios.

In the first two years of the project, we focused on integration scenarios in factory planning. In this field, different specific application programs are used for optimizing machine configuration and transport facilities in factories. The applications are developed independently. Although some kind of computer-based interoperability is necessary, interoperability is currently performed manually. In most cases, interoperation is uni-directional and primarily based on file-exchanges. In consequence, data inconsistencies may occur which means additional work for the factory designer. The result of analyzing the systems is a formal description of the used file formats in SGML and an integrated schema. Beside this, we developed a "hard-coded" demonstration prototype for illustrating problems and features of database integration using federated database systems.

Our second integration scenario was motivated by joint work with our partners in the bio-informatics research group of our institute. In this field, we found again highly heterogeneous databases which contain bio-molecular data that represents results of thirty years of experimental research. The integration of these partly overlapping databases promises a new quality of databases and enables

* This research was partially supported by the German State Sachsen-Anhalt under FKZ: 1987A/0025 and 1987/2527R.

[1] **SIGMA**$_{FDB}$ (**S***chema* **I***ntegration and* **G***lobal integrity* **M***aintenance* **A***pproach for* **F***ederated* **D***ata* **B***ases*)

new applications to access heterogeneous data in a uniform and transparent way. We have to explicitly mention two specific aspects of this scenario. Firstly, most databases are based on files and often provide a WWW interface. Secondly, only reading access is possible. In this context, local (data) autonomy is important and cannot be relaxed. A more detailed discussion about the application of FDBS to this area and a description of the implementation of the prototype **BioBench** can be found in [10].

Our third (and latest) integration scenario is motivated by the *Global-Info* project which aims at supporting digital libraries by federation services. As in the scenarios before, different information systems are available, mainly with WWW interfaces. These systems provide related and overlapping information. At the time being, a user has to access some or many information systems using his personal knowledge to find the correct (and needed) information. For example, even in the computer science society, it is not trivial to assign the right real-world person to the string "M. Scholl". Another problem concerns duplicate representation which is very hard in manual work. For that, a federation layer can support the user in his work with services and meta-data, e.g. author-publication-series-networks.

Our common view to federated database systems is a tightly coupled approach based on the five-level schema architecture of Sheth and Larson [15]. This is motivated by the requirement of data consistency which can only be guaranteed in a tightly coupled environment. A federated database should correctly reflect the modeled real world. Inconsistencies between the databases concerning to same real world entities have to be removed. Correct conflict resolution builds the basis for a correct database integration. Furthermore, we have to consider both intensional and extensional conflicts in a common framework. In doing so, we have to deal with n-ary extentional assertions. Since local integrity constraints determine the semantics of the local schemata, local integrity constraints have to be reflected on the global level. Of course, this approach contradicts the requirement of complete local autonomy. However, from our point of view it is the only way to provide a correct database to users. Even local database users take advantage of a correct database since the data quality will be improved.

Federations build the basis for cooperative systems. In this context, cooperation means working together to achieve a common goal. This was also proven by the first example scenario where consistency and cooperation are the main requirements. On the other hand, we found application scenarios where mainly reading access is possible (and necessary), e.g. in bio-informatics. In this case, a full-fledged FDBS seems to be oversized and/or impractical.

2 Research Fields

The following subsections sketch the research topics on federated databases dealt with in our group.

2.1 Schema Integration

Schema integration is the main step in federated database design. Using the component schemata an integrated schema has to be derived. According to [2],

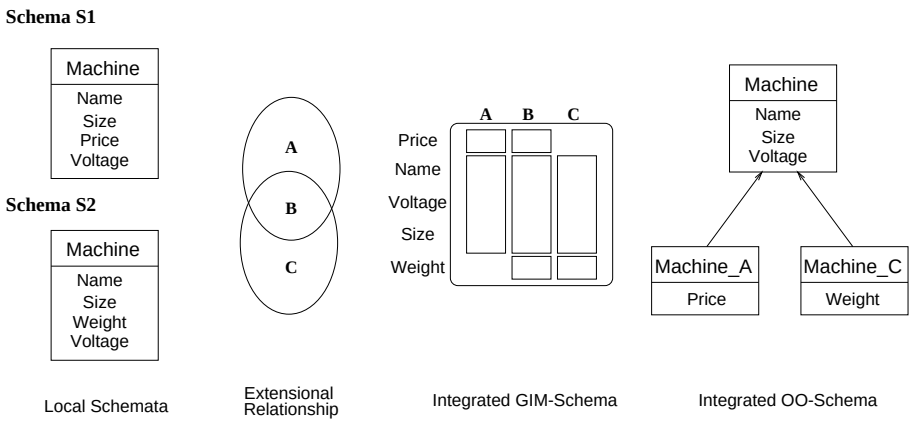

Fig. 1. Schema Integration with GIM

the integrated schema has to fulfill the requirements of completeness, correctness, minimality, and understandability. To achieve this, [11, 12] presented a well-founded methodology based on the data model *GIM (Generic Integration Model)*. The GIM approach provides the basis to systematically handle the possible schema conflicts. In particular, this approach considers extensional conflicts.

In Figure 1 the GIM approach is sketched. One database consist of the data of a world-wide operating company which is selling transport machines. The other database contains machines of a company operating in Europe only. Both component schemata contain a similar class, named Machine. The extensions of these two classes are assumed to be overlapping. This information is usually not available from a local system and has to be modeled manually. Based on this extensional relationships, the schema derivation algorithm computes an integrated GIM schema by extensional decomposition. The resulting schema builds the basis for deriving different views. The corresponding algorithms are implemented in the FDBS design toolkit $\mathbf{SIGMA}_{Bench}$.

2.2 Integrity Constraints and Federated Database Design

Integrity constraints are an important part of database schemata. As pointed out in [3], this often neglected issue has to be included into FDBS design methodologies. A user on the global level needs transparency. That is, the user is not required to have knowledge about the federation or the local databases. The global user only knows the global schema, including global integrity constraints. Only operations, e.g. inserting a new tuple, which are allowed by one of the local systems can be performed. This must be reflected by global integrity constraints. Otherwise an operation could be rejected without a visible reason for the global user. This requirement is named global understandability [17]. An approach to dealing with integrity constraints during schema integration is discussed in [4] for intra-object and uniqueness constraints and in [1] for aggregation constraints.

Another important aspect is the derivation of extensional relationships from integrity constraints. As mentioned before, the definition of extensional rela-

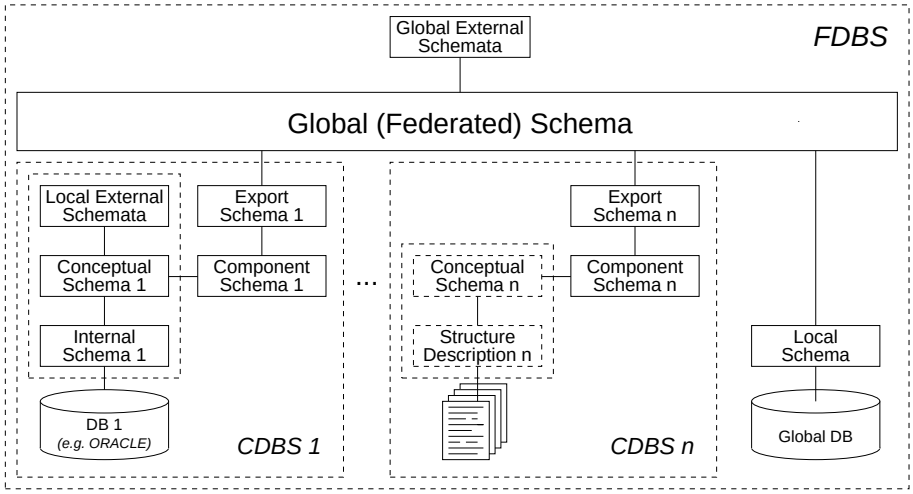

Fig. 2. Extended Schema Architecture

tionships is a quite costly step and has to be done mainly by the designer of the federation. Based on the idea that integrity constraints restrict the possible extension of related classes, the derivation of extensional relationships can be supported [16, 13]. The research results were partially implemented in the **SIGMA**$_{Bench}$.

2.3 File Integration

Many databases containing useful data are based on files. Traditional FDBS design approaches fail in integrating data files since there is neither a data model nor a database schema which can be exploited for schema integration. Nevertheless, the importance of such systems for a federation is confirmed by our analysis of the described application scenarios.

A first approach to integrating file-based systems was presented in [6]. It is important to mention that the integration has to cover all possible files, and should not be restricted to only one or some example files. In this context, we use the term *file cluster* to refer to these file databases which cover all files of an application system. For the integration of such file clusters we suggest the derivation of a structure description, e.g. a grammar, describing the physical structure and the syntax of the data files and a conceptual schema, modeling structure, and semantics in database terms. The structure description could be a SGML-DTD (SGML Document Type Definition) or a context-free grammar according to YACC [6, 7]. The conceptual schema should be based on the common data model of the FDBS (see also Figure 2).

The derivation of these two schemata cannot be done automatically. However, in order to reduce the cost of the very expensive, manual modeling of structure description and conceptual schema as well as to improve the quality of derived wrappers, the process has to be supported by tools. As discussed in [8], the use

of data-mining techniques seems to be very promising. In [10] we deal with the integration of WWW-based data sources which is closely related to file cluster integration.

2.4 Transaction Dependencies in Federated Databases

In a federated database environment, *global transactions* are able to transparently access and manipulate data located in different local database systems. Depending on the underlying global schema, and particularly on the fixed extensional assertions, a global transaction is decomposed into a set of *global subtransactions* which operate on the local database systems. The commits of these transactions must be coordinated. In detail, from the extensional assertions between the classes to be accessed by a global transaction, we derive a set of termination dependencies [14] which sets each corresponding global subtransaction in a special relationship to the global transaction. In summary, different extensional assertions lead to different *global commit rules* which are needed to correctly terminate a global transaction. For instance, in some cases a commit of only one global subtransaction is sufficient to commit the global transaction, i.e., a global transaction can commit although some of the global subtransactions abort [18].

2.5 Security in Federated Databases

A critical feature of database systems concerns the management of confidential data. For that, powerful mechanisms for user management, authorization and authentication are needed. A first but very limited security mechanism is the application of export schemata which support access to only non-confidential data. Of course, in that way, potential advantages of a federation could be lost. Therefore, an FDBS has to provide mechanisms for authorization and authentication which are at least as secure as the mechanisms of the component data management system. Several possibilities to realize a federated authentication component depending on different levels of autonomy and heterogeneity as well as different requirements and environmental conditions are discussed in [5]. We also work on a authorization component. Finally, we investigate a person-oriented security policy for database federations.

3 Prototype Implementation

For demonstration purposes the hard-coded federated system $\mathbf{SIGMA}_{Demo}$ was developed [9]. The prototype demonstrates how a federated system works and how the federation influences local systems. $\mathbf{SIGMA}_{Demo}$ integrates two artificial databases, storing data about conveyor machines. One database is implemented using a relational database management system (Oracle/YARD). The other database uses an object-oriented DBMS (Ontos/ODE). The integrated schema illustrates the solution of the main integration conflicts by applying the GIM approach. Moreover, problems and a solution of integrity constraint integration are demonstrated. The prototype was presented at the CeBIT-Fair 1997.

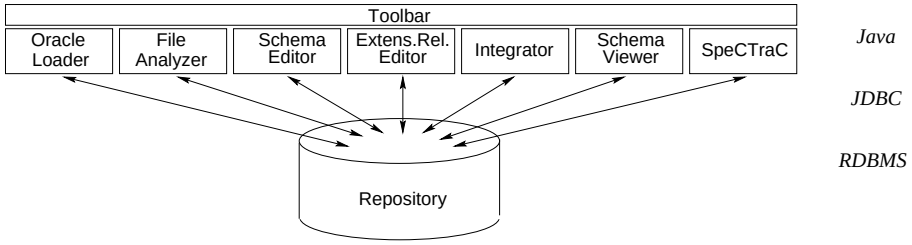

Fig. 3. Architecture of the $\mathbf{SIGMA}_{Bench}$

Since the development of a federated database management system is quite costly, it cannot be done by the limited resources of a research group. Therefore, we focus on the development of federated database (FDB) design methods. Most of the methods are implemented as a part of the integrated FDB-Design-Toolbox $\mathbf{SIGMA}_{Bench}$. The $\mathbf{SIGMA}_{Bench}$ is developed according to the client-server architecture. Central interoperation platform is the $\mathbf{SIGMA}_{Bench}$ repository that is implemented using the relational DBMS YARD (cf. Figure 3). Clients for dedicated tasks are developed in Java and, thereby, provide platform independence. The following clients have been developed or are still in development:

- **$\mathbf{SIGMA}_{Bench}$ Toolbar**
  This application simply provides an easy start-up interface to available $\mathbf{SIGMA}_{Bench}$ tools.
- **Schema Editor**
  The schema editor allows the definition of object-oriented schemata for storing them in the repository. Of course, in "real" integration scenarios the schemata should be imported from the local databases. Therefore, the main issue of the schema editor is to make schemata available for demonstration purposes. Thus, the editor offers beside classes, relationship and specialization also integrity constraints. The schema editor can also be used for enriching imported schemata by integrity constraints.
- **Schema Viewer**
  The schema viewer is used for presenting automatically derived schemata as well as imported and modeled schemata.
- **Extensional Relationship Editor**
  As pointed out, defining extensional relationships is a quite costly manual task. Different approaches to an intuitive graphical user interface were implemented and tested. The current version supports the partial derivation of extensional relationships from integrity constraints and extensional assertions.
- **Integrator**
  The internal part of $\mathbf{SIGMA}_{Bench}$ is the integration tool which automatically derives integrated object-oriented schemata using the source schemata in the repository and modeled extensional relationships. Earlier versions mainly cover extensional and intensional conflicts. The current version provides the integration of integrity constraints, too.

- **Oracle Loader**
  As an example of a loader-tool, an Oracle loader was implemented. It imports schemata from an Oracle database using the catalogue tables of the Oracle system. This includes classes (tables), their attributes, data types, and integrity constraints. After importing an Oracle schema, it can be used for the integration.
- **File Structure Analyzer**
  Different tools for file structure analysis are in development. Key words, tokens, and brackets can be found semi-automatically based on text analysis. In that way, the design of grammars, describing the structure of files clusters, can be supported.
- **SpeCTraC**
  Currently, we are developing the tool SpeCTraC[2] for specifying (global) transactions with different dependencies among them. The tool comprises a consistency checker which supports the designer in specifying consistent dependencies. If an inconsistency occurs, alternative dependencies are proposed. Due to the fixed extensional assertions of the global schema, different commit rules can be derived for global transactions.

4 Conclusions

In this extended abstract, we sketched the research activities and results of the **SIGMA**$_{FDB}$ project. Building and maintaining federated database systems is a hard task which cannot be completely done automatically. Our main goal is to support the schema integrator of a federated database during the design process. We work on different areas of this complex problem and develop different solutions. Currently, we are continuing to implement prototypes to validate our theoretical results.

References

1. S. Balko and C. Türker. Integration of Aggregate Constraints. In *Proc. 2nd Int. Workshop on Engineering Federated Information Systems, EFIS'99*, infix-Verlag, 1999. *Same volume.*
2. C. Batini, M. Lenzerini, and S. B. Navathe. A Comparative Analysis of Methodologies for Database Schema Integration. *ACM Computing Surveys*, 18(4):323–364, 1986.
3. S. Conrad, M. Höding, G. Saake, I. Schmitt, and C. Türker. Schema Integration with Integrity Constraints. In C. Small, P. Douglas, R. Johnson, P. King, and N. Martin, eds., *Advances in Databases, 15th British National Conf. on Databases*, LNCS 1271, pp. 200–214. Springer, 1997.
4. S. Conrad, I. Schmitt, and C. Türker. Considering Integrity Constraints During Federated Database Design. In S. M. Embury, N. J. Fiddian, A. W. Gray, and A. C. Jones, eds., *Advances in Databases, 16th British National Conf. on Databases*, LNCS 1405, pp. 119–133. Springer, 1998.

[2] A Tool for **Spe**cifying **C**onsistent **Tra**nsaction **C**losures

5. E. Hildebrandt and G. Saake. User Authentication in Multidatabase Systems. In R. R. Wagner, editor, *Proc. Ninth Int. Workshop on Database and Expert Systems Applications*, pp. 281–286, IEEE Computer Society Press, 1998.
6. M. Höding. An Approach to Integration of File Based Systems into Database Federations. In *Heterogeneous Information Management, Proc. 10th ERCIM Database Research Group Workshop*, pp. 61–71, ERCIM-96-W003, European Research Consortium for Informatics and Mathematics, 1996.
7. M. Höding. Federating Databases and Files: An Approach to a Uniform and Comfortable Interface to Heterogeneous Tourist Information Systems. In A.-M. Tjoa, editor, *Information and Communications Technologies in Tourism 1997, Proc. Int. Conf.*, pp. 140–149. Springer, 1997.
8. M. Höding and S. Conrad. Data-Mining Tasks in Federated Database Systems Design. In T. Özsu, A. Dogac, and Ö. Ulusoy, eds., *Issues and Applications of Database Technology, Proc. of the 3rd World Conf. on Integrated Design and Process Technology*, Vol. 2, pp. 384–391, Society for Design and Process Science, 1998.
9. M. Höding, G. Grohmann, and E. Hildebrandt. Die FDBS-Demonstrationssoftware **SIGMA**$_{Demo}$. Preprint 3, Fakultät für Informatik, Universität Magdeburg, 1997.
10. M. Höding, R. Hofestädt, G. Saake, and U. Scholz. Schema Derivation for WWW Information Sources and their Integration with Databases in Bioinformatics. In W. Litwin, T. Morzy, and G. Vossen, eds., *Advances in Databases and Information Systems, Proc. Second East-European Symposium*, LNCS 1475, pp. 296–304. Springer, 1998.
11. I. Schmitt. *Schema Integration for the Design of Federated Databases*, Dissertationen zu Datenbanken und Informationssystemen, Vol. 43. infix-Verlag, Sankt Augustin, 1998. (In German).
12. I. Schmitt and G. Saake. Merging Inheritance Hierarchies for Database Integration. In M. Halper, editor, *Proc. of the 3rd IFCIS Int. Conf. on Cooperative Information Systems, CoopIS'98*, pages 322–331. IEEE Computer Society Press, 1998.
13. I. Schmitt and C. Türker. Refining Extensional Relationships and Existence Requirements for Incremental Schema Integration. In G. Gardarin, J. French, N. Pissinou, K. Makki, and L. Bougamin, eds., *Proc. 7th ACM CIKM Int. Conf. on Information and Knowledge Management*, pp. 322–330. ACM Press, 1998.
14. K. Schwarz, C. Türker, and G. Saake. Extending Transaction Closures by N-ary Termination Dependencies. In W. Litwin, T. Morzy, and G. Vossen, eds., *Advances in Databases and Information Systems, Proc. Second East-European Symposium*, LNCS 1475, pp. 131–142. Springer, 1998.
15. A. P. Sheth and J. A. Larson. Federated Database Systems for Managing Distributed, Heterogeneous, and Autonomous Databases. *ACM Computing Surveys*, 22(3):183–236, 1990.
16. C. Türker and G. Saake. Deriving Relationships between Integrity Constraints for Schema Comparison. In W. Litwin, T. Morzy, and G. Vossen, eds., *Advances in Databases and Information Systems, Proc. Second East-European Symposium*, LNCS 1475, pp. 188–199. Springer, 1998.
17. C. Türker and G. Saake. Exploiting Integrity Constraints and Extensional Relationships for Semantic Schema Integration. Preprint 15, Fakultät für Informatik, Universität Magdeburg, 1998.
18. C. Türker, K. Schwarz, and G. Saake. Commit Protocols for Global Transactions in Federated Database Systems. Preprint 2, Fakultät für Informatik, Universität Magdeburg, 1999.