The Indiana Center for Database Systems*

Judith Copler, Director
Indiana Center for Database Systems
Lindley Hall
Bloomington, IN 47401
copler@cs.indiana.edu
(812)855-4890

Executive Directors: Ahmed K. Elmagarmid and Edward L. Robertson

Principal Investigators: A. Bonner[1], R. DeMillo, A. K. Elmagarmid, D. Marinescu, E.L. Robertson, D. Van Gucht

Research Assistants: Jiansan Chen, Jindong Chen, L.S. Colby, W. Du, J. Jing, P. Kalua, Y. Leu, J. Mullen, M. Gandhi, V. Sarathy, A. Zhang

Visiting Scholars and Professors: N. Boudriga (University of Tunis), O. Bukhres (Moorhead State University), M. Gyssens (University of Limburg), E. Kuhn and F. Puntigam (Technical University of Vienna), M. Rusinkiewicz (University of Houston), and L.V. Saxton (University of Regina)

Research Affiliates: C. Davis and D. Shaw (Indiana University), D. Zage and W. Zage (Ball State University)

1. Introduction

The Indiana Center for Database Systems was established in January 1991. A shared research facility with offices at Purdue University and Indiana University, the Center contracts with small database software developers and users of large database systems. Researchers from the Computer Science Departments at IU and Purdue, from the Indiana University School of Business, the School of Library and Information Science and other state universities take an interdisciplinary approach to problems associated with database design, modeling, implementation, and use.

Professor Ed Robertson is the Executive Director of the Center for Indiana University. Professor Ahmed Elmagarmid is the Executive Director at Purdue. Judith Copler serves as the Director of the Center and handles corporate contacts, technology transfer, and educational programs.

Center personnel consult with businesses, large and small, as well as providing various workshops and seminars on database topics. Businesses engage with the Center by providing research topics and opportunities for faculty/ industry scientist exchanges.

The Center has ongoing research projects in a number of areas. This paper summarizes current activities[2].

2. Experimental Prototype InterBase

We have built a prototype multidatabase system called InterBase. InterBase is designed to provide a unified interface to allow users to write global applications over distributed, autonomous and heterogeneous software systems. InterBase runs on interconnected networks consisting of a variety of hosts that include: Sun and NeXT workstations, Sequent machines, IBM mainframes, and IBM/PCs. Our prototype system accepts programs which are specifications of distributed executions written in a language called IPL (InterBase Prototype Language) and uses the IPL interpreter to execute the pro-

*Sponsored by the Indiana Corporation for Science and Technology (CST), a PYI Award from NSF under grant IRI-8857952, grants from the AT&T Foundation, Tektronix, SERC, Mobil Oil and Bell Northern Research.

[1] A. Bonner is now at the University of Toronto.

grams. We have implemented a concurrency controller in InterBase so that we can support multiple simultaneous uses. Currently, we are incorporating the flexible transaction model and a graphical user interface into our prototype. A commitment protocol is under consideration and will be incorporated in our system after theoretical analysis is completed. The implementation of InterBase on Apple Macintoshes is also underway.

[BK91] O. Bukhres and A.K. Elmagarmid, "Interoperability in Multidatabase Systems," In *Encyclopedia Microcomputers*, October 1991.

[REOL90] M. Rusinkiewicz, A. K. Elmagarmid, S. Ostermann, and K. Loa, "The Distributed Operation Language for Specifying Multi-System Applications," In *Proceedings of the First International Conference on Systems Integration*, April 1990.

[EM90] A. K. Elmagarmid and M. Rusinkiewicz, "Critical issues in multidatabase systems," *Information Science*, 1990.

[EC91] A. K. Elmagarmid and J. Chen, "InterBase Prototype: An Execution Environment for Global Applications in Distributed, Heterogeneous Software Systems," CSD-TR-1991.

3. Quasi Serializability

Consistency is a fundamental issue in database systems. It is especially important in heterogeneous distributed database systems because the traditional serializability theory does not work well. In this part of the research, we proposed an alternative, called quasi serializability, which is more suitable to heterogeneous database systems. We studied strengths and weaknesses of quasi serializability and identified restrictions under which quasi serializable executions maintain database consistency. We showed that quasi serializability, unlike serializability, can be maintained without violating autonomy requirements of participating local database systems and developed protocols for ensuring quasi serializability for both replicated and non-replicated heterogeneous distributed database systems.

[DE89] W. Du and A. K. Elmagarmid, "Quasi Serializability: a Correctness Criterion for Global Concurrency Control in InterBase," In *Proceedings of the 15th VLDB Conference*, pages 347-355, August 1989.

[ED91] A. K. Elmagarmid and W. Du, "Integrity Aspects of Quasi Serializability," *Information Processing Letters*, 1991.

[DELO89] W. Du, A. K. Elmagarmid, Y. Leu and S. Ostermann, "Effects of Local Autonomy on Global Concurrency Control in Heterogeneous Distributed Database Systems," In *Proceedings of the Second International Conference on Data and Knowledge Systems for Manufacturing and Engineering*, October 1989.

[ED90] A. K. Elmagarmid and W. Du, "A Paradigm for Concurrency Control in Heterogeneous Distributed Database Systems," In *Proceedings of the Sixth International Conference on Data Engineering*, February 1990.

[DEK91] W. Du, A. K. Elmagarmid and W. Kim", "Maintaining Transaction Consistency in Multidatabases Using Quasi Serializable Executions," In *Proceedings of COMPCON SPRING'91*, 1991.

[DEK91] W. Du, A. K. Elmagarmid and W. Kim, "Maintaining Quasi Serializability in Multidatabase Systems," In *Proceedings of the 7th Interltional Conference on Data Engineering*, April 1991.

4. Transaction Model

Our research effort is focused on formalizing a new transaction model that is suitable for multidatabase systems. The proposed transaction model called *flex* allows capturing more semantics in a transaction. The salient features of *flex* include (1) allowing multiple ways of committing a transaction, (2) controlling the execution of transactions or subtransaction in accordance with external events, and (3) controlling the granularity of isolation of a transaction. We have formalized the *flex* model and proposed two methods for specifying and executing *flex* transactions. We are now implementing these two methods in the InterBase project.

[ELLR90] A. K. Elmagarmid, Y. Leu, W. Litwin, and M. Rusinkiewicz, "A Multidatabase Transaction Model for InterBase," In *Proceedings of the 16th VLDB Conference*, pages 506-517, 1990.

[LEB90] Y. Leu, A. K. Elmagarmid, and N. Boudriga, "Specification and Execution of Transactions for Advanced Databases," Technical Report CSD-TR-1030, Purdue University, October 1990.

[RELL90] M. Rusinkiewicz, A. K. Elmagarmid, Y. Leu and W. Litwin, "Extending the Transaction Model to Capture more Meaning," In *ACM SIGMOD RECORD*, March 1990.

[Le91a] Y. Leu, "Composing Multidatabase Transactions using Flexible Transactions," In *Data Engineering Bulletin*, March 1991.

[Le91b] Y. Leu, "Flexible Transaction Management in the InterBase Project," Ph.D. thesis, Department of Computer Science, Purdue University, August 1991.

[LE89] Y. Leu and A. K. Elmagarmid, "A Hierarchical Approach to Concurrency Control for Multidatabases," In *Second International Symposium on Databases in Parallel and Distributed Systems*, July 1990.

5. Logic Programming

The traditional transaction model is unsuitable for the multidatabase environment. More appropriate transaction models have been developed, such as the *flex* transaction model, however, existing database languages do not support the features of these models. We are currently working on a parallel logic transaction language (PLTL) that supports *flex* transaction model features. A logic language is suitable for our purpose because it provides a framework for concise description of database transactions, tools for reasoning, and supports dependency among these transactions. The design of our language is complete, and its implementation has been started.

[KPE91] E. Kuehn, F. Puntigam, and A. K. Elmagarmid, "Transaction Specification in Multidatabase Systems Based on Parallel Logic Programming," In *Proceedings of the First International Workshop on Interoperability in Multidatabase Systems*, April 1991.

[KELB90] E. Kuhn, A. K. Elmagarmid, Y. Leu, and N. Boudriga, "A Parallel Logic Language for Transaction Specification in Multidatabase Systems," Technical Report CSD-TR-1031, Purdue University, October 1990.

6. Interdependent Data

Many large companies have interdependent data stored in heterogeneous and autonomous database systems. Maintaining consistency of the interdependent data is a real problem faced in many practical computing environments. Unlike replicated data, interdependent data allows the specification of dependency constraints and eventual consistency constraints. The research on maintaining consistency of interdependent data is pursued in two steps. First, we have identified the types of dependencies and the aspects of eventual consistencies of interdependent data. Second, we have presented a new update principle for updating interdependent data. We have also proposed an update control algorithm and an eventual consistency enforcing algorithm for maintaining the consistency of interdependent data.

[SLE91] A. Sheth, Y. Leu, and A. K. Elmagarmid, "Maintaining Consistency of Interdependent Data in Multidatabase Systems," Technical Report CSD-TR-91-016, Purdue University, March 1991.

[EM91] A. K. Elmagarmid and J. G. Mullen, "Negotiating and Supporting Interdependent Data in Very Large Federated Database Systems," Technical Report CSD-TR 91-047, Department of Computer Science, Purdue University, June 1991.

7. Global Transaction Commitment

The issue of global (multi-site) transaction commitment is concerned with getting all the subtransactions of a global transaction to either uniformly commit or uniformly abort, even in the presence of failures. Global transaction commitment is difficult due to the autonomy constraints of local database systems that make up the multidatabase system. For example, it is impossible, in general, to directly implement the two phase commit (2PC) protocol (a simple and common distributed database commitment protocol) in multidatabase systems due to local autonomy constraints. We are in the process of identifying the problems of supporting global commitment and conditions under which it is impossible. We have also defined a new correctness notion called Global Commitment for ensuring global commitment. Global commitment when combined with the correctness notions of recoverability and serializability enables the multidatabase system to guarantee the commitment of global transactions. We have developed a method for supporting the global commitment notion.

[EJK91] A. K. Elmagarmid, J. Jing, and W. Kim, "Global Commitment in Multidatabase Systems," Technical Report CSD-TR-91-017, Purdue University, March 1991.

[ME91] J. G. Mullen and A. K. Elmagarmid, "On the Impossibility of Atomic Commitment in Multidatabase Systems," Technical Report CSD-TR 91-029, Department of Computer Science, Purdue University, April 1991.

8. The Nested Relational Data Model

The nested relational approach is a generalization of the relational approach, obtained by dropping the first-normal-form requirement. This generalization provides a better representation mechanism for complex objects while maintaining the advantages of the relational model. Since attributes of a relation can have relations as values, objects can be represented as a whole in a single nested relation instead of being distributed over several flat relations.

Building on the formal specification of the model, we have conducted theoretical investigations on the design and comparison of various possible query languages. Algebras, obtained by adding programming constructs and by making recursive extensions to the nested relational algebra, besides having practical advantages, provide insight into the expressiveness of query languages and shed light on query optimization techniques.

The principles of a nested relational system are being implemented in ANDA (A Nested Database Architecture). This prototype system follows an innovative approach with separate mappings of values to structural information and of structures to values. Central to its operation is a collection of primitive operations which manipulate these mappings in a primary memory "cache." The high level languages SQL and SQL/NF (an extension of SQL to nested relations) are then implemented on top of cache operations, providing a basis for studying and evaluating query optimization in this model.

[GG88] M. Gyssens and D. Van Gucht, "The Powerset Algebra as a Result of Adding Programming Constructs to the Nested Relational Algebra," In *Proceedings of the 1988 SIGMOD International Conference on Management of Data*, pages 225-232, 1988.

[GG] M. Gyssens and D. Van Gucht, "The Powerset Algebra as a Natural Tool to Handle Nested Database Relations," *Journal of Computer and System Sciences*. To appear.

[GPG89] M. Gyssens, J. Paredaens and D. Van Gucht, "A Uniform Approach towards Handling Atomic and Structured Information in the Nested Relational Database Model," *Journal of the ACM*, Vol. 36, No. 4, pages 790-825, 1989.

[Co89] L.S. Colby, "A Recursive Algebra and Query Optimization for Nested Relations," In *Proceedings of the 1989 SIGMOD International Con-*

ference on Management of Data, pages 273-283, 1989.

[Co90] L.S. Colby, "A Recursive Algebra for Nested Relations," *Information Systems*, Vol. 15, No. 5, pages 567-582, 1990.

[DG88] A. Deshpande and D. Van Gucht, "An Implementation for Nested Relational Databases," In *Proceedings of the Fourteenth International Conference on Very Large Databases*, pages 76-87, 1988.

[De89] A. Deshpande, "An Implementation for Nested Relational Databases," *PhD Thesis*, Indiana University, June 1989.

[GG] M. Gyssens and D. Van Gucht, "A Comparison between Algebraic Query Languages for Flat and Nested Databases," *Theoretical Computer Science*. In press.

[GF87] D. Van Gucht and P. C. Fischer, "Multilevel Nested Relational Structures," *Journal of Computer System Sciences*, Vol. 36, No. 1, pages 77-105, 1987.

9. The List-Structure Data Model

The main goal of this research is to develop suitable modeling and query mechanisms for list-oriented applications. Set-oriented models, like the relational model and the nested relational model, are not well-suited for applications that primarily involve sequential data, such as, scientific experimental data, historical sequences of financial or medical records, and textual data. Some advanced data models provide array-type constructors that can be used to represent sequential data; however, their query languages are not specialized enough to meet the requirements of the types of applications mentioned above.

The list-structure model, like the nested relational model, provides a hierarchical representation mechanism that is suited for complex objects. However, the grouping mechanism in the list-structure model provides for lists rather than sets. The model also provides variable schema representation – a feature that is necessary for applications in which a piece of information can be structured in many different formats. We have developed an algebra for the list-structure model that provides list-oriented functions, such as searching and updating based on patterns and positions of elements, and reordering and sorting elements in a list. In choosing the operators for the algebra, we have tried to maintain a reasonable balance between simplicity and functionality. We are currently investigating various extensions to the model and the algebra (e.g., allowing sets as well as lists). The list-structure model and algebra evolved from the grammar model and some of its associated query languages.

[Co] L.S. Colby, "An Algebra for List-Oriented Applications," To appear as a Technical Report, Indiana University.

[GPG89] M. Gyssens, J. Paredaens, D. Van Gucht, "A Grammar-Based Approach towards Unifying Hierarchical Data Models," In *Proceedings of the 1989 SIGMOD International Conference on Management of Data*, pages 263-272.

10. The Graph-Oriented Object Database Model

Object-oriented database modeling was introduced to deal with data applications such as CAD/CAM, office automation, scientific and engineering databases. There are currently two approaches to developing object-oriented database systems. The first approach extends existing database management systems with object-oriented characteristics. The second approach extends existing object-oriented programming languages with the ability to maintain persistent data effectively.

Our research followed a third approach. We designed an object-oriented database model independent of an existing database system or object-oriented programming language. (This approach gave us maximal freedom in capturing the main features of both solid database modeling and the object-oriented design paradigm.) This database model (the graph-oriented object database model) has a distinctive graphical nature. This was intended to ensure that we have the tools to model applications ranging from simple tabular data, such as is present in conventional databases, to graphical data which arises in CAD/CAM, and to develop a transparent user interface geared towards current workstation environments. Currently, a group of researchers at the University of Antwerp, is building an implementation of this model. In addition, the graph-oriented object database model, has already been proven to be a very useful tool to perform theoretical analysis of the semantics of object-oriented database languages.

Based on the observation that graphs play an important role in the representation of databases (both at the logical and physical level), we introduced an algebra for the manipulation of binary relations, i.e., of directed unlabeled graphs. The algebra is based on early work by Tarski and differs markedly from the (Codd)-relational algebra. (One fundamental difference is that the Tarski algebra is an untyped language. For example, it allows the union of an arbitrary pair of relations, which is clearly not allowed in the relational algebra.) The key notion that we added to Tarski's proposed algebra is tuple tagging. This is needed to give the model enough data modeling and querying power. Tagging can also be seen as a value-based counterpart to object creation in object-oriented data models. We can show that the resulting algebra generalizes the relational algebra, and if further equipped with a looping construct yields a computationally complete database language. Current work is studying the role of tagging in the context of database models which manipulate complex objects.

[GPG90] M. Gyssens, J. Paredaens and D. Van Gucht, "A Graph-Oriented Object Database Model," In *Proceedings of the Ninth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, pages 417-424, 1990.

[GPG90] M. Gyssens, J. Paredaens and D. Van Gucht, "A Graph-Oriented Object Data Model for Database End-User Interfaces," In *Proceedings of the 1990 SIGMOD International Conference on Management of Data*, pages 24-33, 1990.

11. A Framework for Representing Data Models

Several data models, including the data models mentioned above, have been designed to meet the needs of complex data applications that traditional data models are incapable of supporting. Unfortunately, the data models are all defined using different formalisms with the result that it is difficult to compare these models and understand the underlying similarities and differences in them. We are investigating the use of first-order logic as a unifying framework for representing data models. One of our principal goals is to use this framework for studying the semantic equivalence of different data models.

12. Relational Query Processing via Complex Objects

Our recent work on algorithms for complex objects indicates that nested relations may enhance efficiency, contrary to the conventional wisdom. For instance, the original definition of SEQUEL had a CONTAINS operator which was dropped largely for implementation concerns, but this is exactly the kind of operator that may enhance efficiency in certain situations. The particular impact of these new algorithms is in non-monotonic queries, i.e., queries where the size of the answer may decrease as information is added to the database (the answer to "give me all employees with no dependents" gets smaller when Jones gets married). The nested relational algebra (the extension of the relational algebra to nested relations) allows the succinct expression of non-monotonic queries and our algorithms provide efficient evaluation of these expressions.

Our current efforts in this area include the implementation of these algorithms on top of a traditional relational database (INGRES) in order to provide timing comparisons. Also, we are exploring the translation of flat relational queries into the nested formalism in order to take advantage of the efficiencies that we have discovered.

[SG90] L.V. Saxton and D. Van Gucht, "Optimization of Nonmonotonic Relational Queries," Technical Report 322, Indiana University, December 1990.

[PG] J. Paredaens and D. Van Gucht, "Converting Nested Algebra Expressions into Flat Algebra Expressions," *ACM Transactions on Database Systems*. To Appear.

13. Temporal Databases

Because the issues of temporal databases are so much those of user semantics, an important part of our study is to begin with the investigation of user needs. To this end, we have been interacting with natural and social scientists to understand their needs for representing and manipulating temporal data. For example, social historians are studying family behavior over several generations, asking questions about sequences of "life-events", such as "does living with her grandchildren younger than five years tend to prolong the life of an elderly widow?" Beyond merely providing a convenient way for users to specify time "windows," it is necessary to facilitate querying time-sequence behavior.

The natural relationship between nested relations and historical data (an attribute represents a set of values and their valid times) has led to extensions to the ANDA system specifically for storing and manipulating temporal information. This implementation uses the nested model to provide the semantics but adds special structures to avoid serious inefficiencies that would be imposed by the "dispersed" nature of the ANDA storage structure. In addition, new time index structures will be required if "snapshot" queries ("what was the state of the world at time *T*?") are common occurrences – a further cost which must be carefully considered.

[KR91] P. Kalua and E.L. Robertson, "The Role of Time in Information Systems," In *Proceedings of the Second International Conference for Computing in Southern Africa Foundation (CISNA-91)*, April 1991.

14. Specification Based Models

Work is underway to develop a data model that provides a uniform framework for software development, VLSI design and manufacturing, mechanical CAD/CAM and other applications with a parts-and-components structure. Key to this model is the distinction between specification, design, and manifestation. The recognition that "version" is a local property while "configuration" is global provides version control without version proliferation and naturally incorporates testing in the design and development process.

---

[2] Copies of research reports or additional information on specific researchers and their projects can be obtained from the Center:

Judith Copler, Director
Indiana Center for Database Systems
Lindley Hall
Bloomington, IN 47401
copler@cs.indiana.edu
(812) 855-4890

or

Sue Barton
Indiana Center for Database Systems
Department of Computer Science
West Lafayette, IN 47907
(317)-494-3301
barton@cs.purdue.edu