Panel on Extensible Database Systems

D.S Batory, Department of Computer Sciences, The University of Texas at Austin
M. Mannino, Department of General Business, The University of Texas at Austin

Panelists:

M. Carey, Computer Sciences Department, University of Wisconsin at Madison
U. Dayal, Computer Corporation of America
C. Mohan, IBM Almaden Research Center
L Rowe, Computer Science Division - EECS, University of California, Berkeley

Background and Motivation

New implementation techniques and new capabilities for database systems are being developed and proposed at a rapid rate Novel file structures and improved algorithms for query optimization, buffer and recovery management, and transaction management have the potential of realizing significant gains in DBMS performance The proposed integration of design objects, voice, text, rules, vector graphics, and images into databases promises exciting new capabilities for DBMSs To accommodate advances in database technology and to support new classes of database applications, DBMSs must be extensible (i e , customizable)

To achieve extensibility forces a fundamental rethinking about how DBMSs are built, and how special-purpose features can be integrated into a DBMS with little effort and expense Customizing DBMSs implies the availability of extensible data models, to allow for the introduction of new object types and operations, and extensible storage structures, to take advantage of special properties of stored data or operations to enhance performance

Although research on extensible DBMSs is still in its infancy, a fundamental concept underlying their construction is now evident This is the standardization of interfaces and the plug-compatibility of modules An extensible DBMS will be a 'software bus' whereby new modules (and hence new DBMS capabilities) can be added, exchanged, or removed by plugging or unplugging modules Extensible DBMSs will thus rely on extensive software libraries, where new modules can be added as needed Furthermore, changes to DBMSs can be made in months rather than years, and the reinvention of established technology is kept to a minimum because of the reusability of modules

The perception of DBMSs as monolithic entities that are difficult to modify will change as extensible DBMS technology becomes better understood The use of database systems will not change, the ANSI/SPARC roles of database users, who write and execute transactions, and the database administrator (DBA), who designs and writes database schemas, will remain Extensible DBMSs will require the introduction of an additional party, the *database architecture administrator (DAA)*, who is responsible for the construction and customization of a DBMS

A growing number of researchers are developing extensible DBMSs The purpose of this panel is to explain and discuss some of the approaches that are now being taken (and those that can be taken), and to survey the problems that confront extensible database technology Descriptions of the systems and research represented at this panel are given in the following sections

The EXODUS Project [1,2]

M Carey, D. DeWitt, D Frank, G Graefe,
M Muralikrishna, J. Richardson, and E. Shekita

Computer Sciences Department
University of Wisconsin, Madison, WI 53706

Until recently, research and development efforts in the database management systems area have focused on supporting traditional business applications The design of database systems capable of supporting non-traditional application areas, including engineering applications for CAD/CAM and VLSI data, scientific and statistical applications, expert database systems, and image/voice applications, has emerged as an important new direction for database system research These new applications differ from conventional applications such as transaction processing and from each other in a number of important ways First, each requires a different set of data modeling tools The types of entities and relationships that must be described for a VLSI circuit design are quite different from the data modeling requirements of a banking application Second, each new application area has a specialized set of operations that must be supported by the database system It makes little sense to talk about doing joins between satellite images Efficient support for the specialized operations of each of these new application areas requires new types of storage structures and access methods as well Access and manipulation of VLSI databases is facilitated by new access methods such as R-Trees [Gut84] Storage of image data is simplified if the database system supports large multidimensional arrays as a basic data type (a capability provided by no commercial database system at this time) Storing images as tuples in a relational database system is generally either impossible or terribly inefficient. Finally, a number of these new application areas need support for multiple versions of entities [Day85, Kat86] To address the needs of this emerging class of applications, we are working on the design and implementation of an extensible database system known as EXODUS [1] at the University of Wisconsin [Car85]

[1] EXODUS A departure, in this case from the ways of the past Also an EXtensible Object-oriented Database System.

[2] This research is partially supported by the Defense Advanced Research Projects Agency under contract N00014-85-K-0788, by the National Science Foundation under DCR-8402818, and by an IBM Faculty Development Award.

In contrast to some of the related work in this area, EXODUS is being designed as a modular (and modifiable) system rather than as a 'complete' database system intended to handle all new application areas The EXODUS storage system [Car86] is the kernel of the system Since it is to be the only fixed component of the EXODUS system, its design is intended to be flexible enough to support the needs of a wide range of potential applications Application-specific access methods, operations, and version management layers will be constructed using the primitives provided by the storage system, and higher levels of the system will in turn use the primitives supplied by these layers The EXODUS design includes a generic query optimizer that optimizes a generalized algebraic query tree based on a collection of cost and operator transformation rules that implementors of application-specific DBMS's will provide At the top level, EXODUS will provide facilities for generating application-specific high-level query language interfaces, but applications will also be permitted to interact with the system at lower levels when necessary Thus, the EXODUS approach might be characterized as the 'DBMS generator' approach, with the overall goal of the project being to implement the storage system, the tools to support development of appropriate abstract data types, access methods, operations, and version support, the rule-based optimizer; and the flexible query interface generator To aid application-specific DBMS developers in their task, we also expect to provide libraries of useful routines (and rules) for the extensible components of the system

STARBURST An Extensible Relational DBMS

C Mohan

IBM Almaden Research Center
San Jose, CA 95120

The STARBURST project's goal is to produce a portable, extensible, distributed database management system for the 1990's, building on our previous experiences to invent the required state-of-the-art technology We will be re-examining the approaches taken to query optimization, access paths, transaction management, and other areas in earlier database management systems, and designing an 'open database architecture' that will allow knowledgeable users to tailor the DBMS to support their particular applications Such users may, for example, define new data types, new operations, new methods to access data stored inside or outside the DBMS, and new strategies for concurrency control and recovery

The focus will be on the database capability needed for departmental servers, in order to support a wide range of applications and to couple effectively to mainframes, other servers, and private workstations As far as distributed data management is concerned, the focus will be on 'vertical distribution' - that is, the coupling between host and server and server and workstation Compared to R*, vertical distribution is expected to impact differently issues like naming of different entities, distribution of query optimization work, authorization, and recovery The changes are anticipated due to the disparities in processor and I/O capacities, the level of trustworthiness of the protection environment, communication bandwidths, and data sharing requirements

The immediate objective of the STARBURST project, which was formed about a year ago, is to get a centralized base system running so that it will serve as the basis for the extensibility and distribution work To facilitate portability, it is being implemented in C Compared to System R, we are currently exploring different approaches to record management, index management, long fields, query compilation, and recovery We are also exploring alternative approaches to finding the best strategy for query execution New algorithms to provide record level locking with write-ahead logging and recovery are being designed Some of these changes relate to our objective of improving the state of the art. The major goal of extensibility is affecting many parts of the system In the area of query optimization, we are considering ways to improve the optimizer's cost models and to support user control of the extent of query optimization, especially for ad hoc queries In the query language area, attention is being given to providing tree structured responses, composition of new data types from already existing types, and user-defined scalar and aggregate functions Query compilation is being restructured for better modularity and extensibility Mechanisms for supporting extension-managed access paths and data are also being investigated

There are a number of open issues in the areas we are currently studying and in the ones we will be addressing later Many of them relate to extensibility and vertical distribution Some of the issues have to do with questions like how to integrate concurrency control and recovery of user-managed data with DBMS managed data, how to limit the damage that could be caused by user extensions, how to inform the query compilation component about user-managed data and access paths, how to let the optimizer get to know the costs and selectivities associated with user-managed data and access paths, and user-defined operations The next phase of the project should provide at least partial answers to some of these questions

GENESIS [3]

D S. Batory

Department of Computer Sciences
The University of Texas, Austin, TX 78712

In the last five years, theories of database system implementation were developed to explain the storage architectures of many commercial DBMSs (i e , how these systems stored and retrieved data) These theories identified basic components of DBMS software, required all components to have the same interface, and showed that component composition could be achieved in a simple manner [Mar81, Bat82, Bat85b]

GENESIS is an extensible DBMS that is based on these theories A prototype is now operational [Bat86] The software library on which GENESIS relies contains modules that implement the components of these theories These components are simple files (file structures), linksets (record linking structures), and elementary transformations (conceptual-to-internal mappings) The file management system of GENESIS, called JUPITER, features a number of single-key and multi-keyed file structures Among the structures presently supported are indexed-sequential, indexed-aggregate, B+ tree, unordered, hash-based, heap, and multi-key hash structures JUPITER can be reconfigured to handle any number of file structures, in addition to supporting recovery via shadowing, recovery via database cache, or no recovery at all

Once a storage architecture has been designed, only the modules that are not present in the library must be written As all modules are reusable, we anticipate the need for adding new modules will decrease as the library enlarges When all modules are present, the time it takes to write the specification (a short C program) and to reconfigure GENESIS is a matter of hours, and can be done with negligible cost

The GENESIS prototype presently features a DDL and a procedural DML for non-first-normal-form relations Tuples can have long fields, variable-length fields, multidimensional matrices, and repeating and nested repeating fields (i e , relation-valued attributes) The future DDL/DML of GENESIS will be based on a functional data model/data language, which will reference a library of data types and operators The functional front-end to GENESIS should be operational in 1987

[3] This research is partially supported by the National Science Foundation under MCS-8317353

A primary goal of GENESIS is to help consolidate theoretical results and practical achievements in database research by providing a pragmatic and encompassing theory of DBMS implementation Among the open problems that remain to be integrated into the prototype are concurrency control, query processing, and design objects [Bat85a] The layered approach that is required to specify DBMS storage architectures forces a novel interpretation and generalization of existing results on these topics

PROBE A Research Project in Knowledge-Directed Database Management

U Dayal

Computer Corporation of America
Four Cambridge Center, Cambridge, MA 02142

Conventional record-based DBMSs will be inadequate for many of the knowledge-intensive information processing applications (e g , business and industrial automation, CAD/CAM, and military command and control) of the future These applications require integrated access to a variety of information types (e g , images, maps, signals, text) not currently supported by DBMSs Also, they rely on specialized knowledge or expertise for processing the new information types, for many of these types, specialized storage devices and processors (e g , workstations, image enhancers, solid modellers), are or will be available Currently, DBMSs have no general facilities for efficiently assimilating and utilizing this special knowledge or for incorporating these specialized processors into their own processing

The objective of the PROBE project is to develop an advanced DBMS effective for these knowledge-intensive applications Our approach is to enhance existing DBMSs with (a) user-defined object classes as the basis for defining new information types and operations and for integrating specialized processors, (b) dimensional (space and time) concepts, which are a common characteristic of many of the new information types, and (c) recursive predicates and queries, which provide intensional knowledge processing capabilities essential for many of the applications In each case, it is necessary to augment both the logical (data model, query language) components and the physical (storage structures, access methods, query processor) components of the DBMS Overviews of PROBE are contained in [Day85a, Day85b] The PROBE approach to efficiently processing recursive queries is described in [Ros86], and dimensional queries in [Ore86]

Because the PROBE DBMS is object-based, it is extensible new object classes can be easily defined However, unlike the proposals to add abstract data types to DBMSs as 'black boxes' that hide all implementation detail, the PROBE approach is to reveal some information about the properties and costs of the class's operations to the global query optimizer to enable it to construct efficient global execution plans The design of an extensible, description-driven query optimizer is an important open research problem The successful solution of this problem will mean that extensibility can be attained without sacrificing performance

Extensible Query Optimization

M Mannino

Department of General Business
The University of Texas, Austin, TX 78712

Despite all the research on query optimization in the last ten years, constructing an optimizer even for an established language such as SQL is still a formidable engineering task The reasons for this difficulty are the large, complex nature of a query optimizer and the research focus over the last ten years Most of the research involves larger and more complex domains such as distributed systems, recursive queries, and multiple query optimization The engineering difficulties in constructing a query optimizer have been largely ignored

Research in extensible query optimization seeks to address these engineering concerns Two major objectives of extensible query optimization are 1) to understand the building blocks of query optimizers, and 2) to reduce the time and effort to construct, maintain, and extend an optimizer One way to approach the first objective is to decompose the problem into a collection of expert modules and to explicitly define the rules, data structures, interface and inference procedure employed in each module The emphasis in defining the expert modules is to simplify and generalize the knowledge used in the optimization process This requires representation schemes that serve a wide variety of languages, access methods, and optimization strategies

The second objective is related to the first Certainly, if our understanding of the process improves, we will be able to construct, maintain, and extend optimizers more efficiently However, to achieve significant reductions, the manner of designing and implementing optimizers must change Two approaches seem promising In the knowledge-driven approach, the cost formulas, selectivity estimation techniques, matching rules, levels of optimization, and search strategies are encoded outside of the optimizer's source code The task of extending the optimizer with new access methods, data types, comparison operators, join algorithms, and search strategies can be accomplished by altering the external descriptions rather than the optimizer's source code

In the tool approach, an optimizer is constructed by using tools such as general purpose libraries and specification driven tools The libraries permit an optimizer to be built by combining the procedures in new ways The procedures must be general to support many optimization environments A specification driven tool produces a customized program from a precise specification Such tools are analogous to traditional parsing and lexical analysis tools As an example, a selectivity estimation tool would use a data type description including the types's internal representation, comparison operators, and selectivity estimation rules to produce a program to estimate the selectivity of a relational expression involving the data type

Most of the early work on extensible query optimization addresses the knowledge-driven approach because it is the easiest to cost justify for an established database vendor The research on the tool approach is easier to justify for a group that does not have vested interest in a database product In the future, both approaches will be important to reduce the cost and effort of building, maintaining, and extending query optimizers

POSTGRES

L A Rowe

Computer Science Division - EECS
University of California, Berkeley, CA 94720

POSTGRES is a new database management system that is being developed at Berkeley as a successor to the INGRES relational database system The design goals for the system are to

- support for complex objects,
- make the system extensible,
- support active databases (i e , alerters and triggers) and inferencing,
- simplify the DBMS code for crash recovery,
- and to design the system to take advantage of optical disks, workstations composed of multiple tightly-coupled processors, and custom designed VLSI chips

The system will use a conventional relational data model However, the program interface will be based on 'portals', rather than 'cursors', which will make the development of interactive browsing style applications easier More details on the design and proposed implementation are given in another paper in the proceedings [Sto86]

POSTGRES will allow users to extend the system by defining new data types, operators, built-in functions (including aggregates), and access methods All data types and operators will be defined by using the data type and operator extensibility features of the system Scalar functions can be called in a query or they may be dynamically linked into a front-end program so that user applications will be able to manipulate the data using the same abstraction supported by the DBMS Finally, new access methods can be defined and used either for storing primary data or secondary indexes The specification of an access method includes information about the operators in queries that can be optimized by using it

The POSTGRES implementation is currently being designed and coding has begun The first working prototype will be completed during the middle of 1987

References

[Bat82] D S Batory and C C Gotlieb, 'A Unifying Model of Physical Databases', *ACM Trans Database Syst*, 7,4 (Dec 1982), 509-539

[Bat85a] D S Batory and W Kim, 'Modeling Concepts for VLSI CAD Objects', *ACM Trans Database Syst*, 10,3 (Sept. 1985), 322-346

[Bat85b] D S Batory, 'Modeling the Storage Architectures of Commercial Database Systems', *ACM Trans Database Syst*, 10,4 (Dec 1985), 463-528

[Bat86] D S Batory, J R Barnett, J F Garza, K P Smith, K Tsukuda, B C Twichell, and T E Wise, 'GENESIS A Reconfigurable Database Management System', Tech Rep 86-07, Department of Computer Sciences, The University of Texas at Austin, 1986

[Car85] M Carey and D DeWitt, 'Extensible Database Systems', *Proceedings of the Islamorada Workshop on Large Scale Knowledge Base and Reasoning Systems*, Febuary 1985

[Car86] M Carey, D DeWitt, J Richardson, and E Shekita, 'Object and File Management in the EXODUS Extensible Database System', Technical Report, Computer Sciences Department, University of Wiconsin-Madison, March 1986

[Day85a] U Dayal and J Smith, 'PROBE A Knowledge-Oriented Database Management System', *Proceedings of the Islamorada Workshop on Large Scale Knowledge Base and Reasoning Systems*, February 1985 Also, in M Brodie and J Mylopoulos (eds), *On Knowledge Base Management Integrating Artificial Intelligence and Database Technologies*, Springer-Verlag, 1986

[Day85b] U Dayal, A Buchmann, D Goldhirsch, S Heiler, F Manola, J Orenstein, and A Rosenthal, 'PROBE - A Research Project in Knowledge-Directed Database Management Preliminary Analysis', Technical Report, CCA-85-03, July 1985

[Gut84] A Guttman, 'R-Trees A Dynamic Index Structure for Spatial Searching', *ACM SIGMOD 1984*, 47-57

[Kat86] R Katz, E Chang, and R Bhateja, 'Version Modeling Concepts for Computer-Aided Design Databases', *ACM SIGMOD 1986*

[Mar81] S T March, D G Severance, and M Wilens, 'Frame Memory A Storage Architecture to Support Rapid Design and Implementation of Efficient Databases', *ACM Trans Database Syst*, 6,3 (Sept 1981), 441-463

[Ore86] J Orenstein, 'Spatial Query Processing in an Object-Oriented Database System', *ACM SIGMOD 1986*

[Ros86] A Rosenthal, S Heiler, U Dayal, and F Manola, 'Traversal Recursion A Practical Approach to Supporting Recursive Applications', *ACM SIGMOD 1986*

[Sto86] M Stonebraker and L Rowe 'The Design of POSTGRES', *ACM SIGMOD 1986*