To better understand the nature of heterogeneous distributed database systems, let’s consider the following scenario of a large automobile manufacturer whose operation relies on such databases as

1. **Design.** A collection of part geometry and part features for cars, pickup trucks, and vans.
2. **Process planning.** A hierarchy of alternative sequences of operations for fabricating specific parts of a car such as the body, seats, and engine; robot programs; numerical control programs; inspection programs; and kitting instructions for materials packaging.
3. **Resource planning.** Classes and instances of systems in facility, location, allocation, and usage schedules.
4. **Work in process.** Orders, work orders, parts inventory, workpiece status, and tray/carrier status.
5. **Tooling.** Type, location, status, and remaining lifetime of all portable tools, fixtures, and end-effectors.
6. **Machine.** Current location of mobile equipment, status, and time in process of current machining operations, coolant levels, contents of the tool changer, etc.
7. **Finished products.** Inventory of finished products, due dates for availability of models, description of each model, etc.

Let’s assume that, in addition to containing a number of diverse elements, each database is a different type. For instance, design data may reside in an object-oriented database, machine data in a relational database such as an IBM DB2, and tooling data in a hierarchical database such as an IBM IMS (information management system).

Now, let’s consider the question, When will a new automobile model be available if the designs of components 12345, 87654, and 76548 are modified? Design changes in a part require fabrication changes and the allocation of machines to fabricate that part. Modifying a design changes the manufacturing schedule, inventory, and availability of the products that use that part. To answer the question, the user would have to access more than one database. Since each database uses a different language, model, and access technique, answering this question is no simple matter.

A heterogeneous distributed database system (HDDS) could help by analyzing the question, identifying the databases required to answer it, fetching the information, assembling the results, and presenting them to the user. Ideally, all this would be done transparently.

A major challenge of integrating diverse databases is hiding the heterogeneity of the constituent databases from users. In theory, an HDDS should preserve the autonomy of constituent data-
bases. This implies that the HDDS should neither impose changes on existing databases nor require any reprogramming of the local database management systems (DBMSs). The system should appear as a single integrated database. This includes hiding the heterogeneity of file systems, data models, database languages, and data semantics, as well as the hardware and operating systems on which the databases run. Further, the masking of heterogeneity should add a minimal overhead to processing time and the consequent response time.

Increased processing time can occur in two ways. Queries must be translated into a form that each database system understands. In addition, the results obtained from each system have to be interpreted, assembled, and presented to the user. In practice, many of these objectives are extremely hard to achieve (see sidebar).

Developing HDDSs

The two major approaches for establishing an HDDS from separate databases are a unified schema and a multi-database. Proponents of the first approach advocate establishing an integrating model to define a unified schema of the constituent databases. This schema is also called global. The model used for defining this schema must be a superset of the underlying database models. All transactions (queries and updates) requiring access to more than one underlying database have to transpire through the global schema.

The multidatabase approach has no single integrated schema. Advocates of this approach argue that complete integration is not necessary to preserve the autonomy of the constituent databases. Each database continues to operate in an independent manner. However, each system also forms a part of a federation of users who can share information. This may occur in a scientific community that shares an extremely large number of databases.

Definition of a single global schema would be problematic and even unnecessary. The central questions in this case are, What degree of sharing should be allowed, and How should this be managed? Most research in this area has

Challenges in a heterogeneous database environment

Centralized databases were predominant during the seventies. This decade also saw the advent of popular commercial database management systems based on relational, hierarchical, and network models. Since each model was suited for different applications, many diverse DBMSs developed. An HDDS is required to access these diverse databases in a unified manner.

An HDDS must support preexisting databases without requiring them to undergo conversions or major modifications. The reason for this is economy. Major changes in the databases would necessitate major — and prohibitively expensive — changes in the software. Clearly, certain changes in DBMSs will be needed to accommodate standard interchange protocols, for example, but the effects of such changes on existing programs should be minimal. Developing an HDDS poses a number of interesting challenges and research questions.

Definition of an integrating model. A critical requirement of an HDDS is the development of a strong integrating model. This model should have sufficient power to capture the conceptual relationships among the information units and the objects in the databases. Such power is necessary to express the various relationships and semantic information captured by different data systems. Several "semantic" models have been developed to serve as the integrating model. Most of these models incorporate object-oriented constructs. In our manufacturing example, information such as part geometry, tooling data, and inventory would be described using this integrating model.

Schema integration. Once researchers construct a strong integrating model, they still have the problem of defining each underlying database (or local database) to obtain a unified schema. Semantic differences such as synonyms, homonyms, naming conflicts, and differences in attribute formats and field lengths need resolution. Different databases may pose varying integrity constraints such as rules for existence dependencies or an allowed range of values for different fields. Any conflicts in these areas also need to be resolved before a unified schema can be defined. An interesting challenge here is to develop automated tools to help integrate the schema.

Mapping methodology. Once a given schema is defined, researchers must focus on the problem of mapping this definition to the underlying databases. Given a specific information model and a database that implements it, one can always relate the database constructs to those of the model. The problem is to devise a "language" in which such relationships can be expressed. The language must be sufficiently exact so that some form of it can be used by a distributed data system to map operations from the modeled information into operations on a corresponding database. The language also should be sufficiently powerful to describe most reasonable implementations of an arbitrary instance of the information model. The mapping language must therefore support relational, hierarchical, navigational, and object-oriented database organization.

Data administration functions. Data administration in an HDDS involves processing transactions efficiently and effectively. This is a particularly challenging problem. The key issues here are concurrency control and recovery. Concurrency control techniques should ensure that the underlying databases remain consistent in spite of concurrent accesses. The existence of a large number of concurrency control techniques complicates this problem. Each DBMS may be using a different concurrency control technique (such as locking or time-stamping). The global manager should arbitrate among global and local transactions to ensure their proper execution. Recovery techniques in a heterogeneous database environment are also correspondingly more complicated because each affected database must be restored to a consistent state after a crash.
resulted in systems that support queries, although some systems provide limited updating.

**Standards: A panacea?**

Many believe that standards development will resolve problems inherent in integrating heterogeneous databases. The idea is to develop systems that use the same standard model, language, and techniques to facilitate concurrent access to databases, recovery from failures, and data administration functions. This is easier said than done. Agreement on standards has proven to be one of the most difficult problems in the industry. Most vendors and end users have already invested in separate solutions for their problems. Getting them to agree on a common way of handling their data is challenging.

Heterogeneity also arises out of the diverse needs of applications and company mergers or acquisitions. New applications also produce heterogeneity. Developing standards for heterogeneous databases understandably requires considerable experience with implemented systems. We are just now beginning to understand the issues.

The International Standards Organization (ISO) and the American National Standards Institute (ANSI) are active in this area. The ISO has drafted the Remote Database Access (RDA) standard to provide a single interface for heterogeneous databases. RDA is based on a client/server architecture and uses the Open Systems Interconnection (OSI) model.

The generic RDA standard can be refined to support specializations for use with specific data models such as relational. To help expedite the development of implementations based on this standard, more than 40 vendors and users have established the Structured Query Language Access Group for specializing the RDA for SQL systems.

A heterogeneous database environment poses many interesting research challenges. In this issue of *Computer*, we have tried to highlight some of these problems and their solutions. Many corporations, research institutions, and universities are working to resolve these problems. We hope this issue will help.
Work on the frontier of computational research.

Bring your technical experience to Los Alamos National Laboratory where we have challenged the frontiers of science, researching an exciting range of breakthrough discoveries in many areas.

Postdoctoral Candidates Software Development

We are currently accepting applications for a postdoctoral or postgraduate candidate to conduct R&D activities in the areas of multi-user, client-server databases and graphical user interfaces. The successful candidate will work as part of a small software configuration management team in support of critical Department of Energy software projects. The candidate will assume primary responsibility for the design, implementation and commissioning of a state-of-the-art, X-Windows-based, multi-user database that integrates the group’s SCM activities. In addition, the successful candidate will contribute to the formal program of software quality assurance operated by the group.

Applicants should hold a Master’s or Doctoral Degree in Computer Science, Software Engineering or related discipline. Applicants must demonstrate significant experience in the areas of relational database systems and user interface design. Experience with X-Windows application development is desirable. Familiarity with large-scale software development projects, formal software engineering methods, and software configuration management is desirable.

To formally apply for this position, interested candidates should forward a resume, three external letters of reference, undergraduate and graduate transcripts to Mary Anne With (MS P280), Human Resources Development Division PD-91-094, Los Alamos National Laboratory, Los Alamos, NM 87545. Affirmative Action/Equal Opportunity Employer. Must be able to obtain a Department of Energy Security Clearance.

Los Alamos
NATIONAL LABORATORY

Acknowledgments

I thank Bruce Shriver, former editor-in-chief of Computer, for his enthusiastic support and encouragement of this special issue; Jon Butler, the current editor-in-chief, for his invaluable help in gathering reviews and processing the manuscripts; Carla Shaw, Marwan Aiche, and Kelly O’Neill for their assistance with the manuscripts; and all reviewers and authors whose contributions made this issue possible.

For further information

HDDS research has been receiving increasing attention in the past few years. A number of conferences and journals have focused attention on this topic. Some of the important topics in this area have been covered in this issue of Computer. Interested readers may also refer to the following list.


Sudha Ram is an associate professor of management information systems at the University of Arizona. She is working on research projects funded by IBM, NCR, the US Army, the National Institute of Standards and Technology (NIST), and the Marketing Science Institute. Her research interests are database and knowledge-based systems.

Ram received a BS degree in mathematics, physics, and chemistry from the University of Madras in 1979, a P.O.D.M. from the Indian Institute of Management, Calcutta, in 1981, and a PhD from the University of Illinois at Urbana-Champaign in 1985. She has written articles published in such journals as Communications of the ACM, IEEE Expert, and IEEE Transactions on Knowledge and Data Engineering. She is a member of the editorial board of the Journal of Database Administration and a member of the ACM, the IEEE Computer Society, and the Institute for Management Science (TIMS).

Readers may contact Sudha Ram at the Department of Management Information Services, College of BPA, University of Arizona, Tucson, AZ 85721.