Remote-Exchange: An Approach to Controlled Sharing among Autonomous, Heterogeneous Database Systems

Douglas Fang, Joachim Hammer, Dennis McLeod, Antonio Si

Computer Science Department
University of Southern California
Los Angeles, CA 90089–0782

Abstract

This short paper describes several aspects of the **Remote–Exchange** project at USC, which focuses on an approach and experimental system for the controlled sharing and exchange of information among autonomous, heterogeneous database systems. The spectrum of heterogeneity which may exist among the components in a federation of database systems is examined, and an approach to accommodating such heterogeneity is described. An overview of the **Remote–Exchange** experimental system is provided, including the top level architecture, sharing mechanism, and sharing "advisor".

1 Introduction

Consider an environment which consists of a collection of data/knowledge bases and their supporting systems, and in which it is desired to accommodate the controlled sharing and exchange of information among the collection. We shall refer to this as the (interconnected) autonomous heterogeneous database environment. Such environments are extremely common in various application domains, including office information systems, computer-integrated manufacturing systems (with computer-aided design as a subset), personal computing, business and financial computing, and scientific research information bases. The trend towards the decentralization of computing that has occurred over the past decade has accentuated the need for effective principles, techniques, and mechanisms to support the information sharing and exchange among the component data/knowledge base systems, while maximally retaining autonomy for the components.

Traditional research on "distributed databases" (see, e.g., [CP84]), assumed a common, integrated database specification (conceptual database schema). While some of the research results obtained in this general area of endeavor are applicable in the autonomous heterogeneous database environment, such approaches generally assume a single conceptual database which is physically distributed. Work on "multi-databases", "superviews", and "virtual databases" has stressed the need to provide a unified, perhaps partial, global view of a collection of existing databases [DH84, LA86, MB81]. Techniques for database integration [BLN86], which are often primarily considered for the design of a single database system based upon a number of application subsystems, can also be brought to bear on the problem of partially integrating existing heterogeneous databases. Notably, techniques for multi-databases and database integration all focus on conceptual schema level diversity. "Federated database" architectural issues and techniques for information sharing at the conceptual schema level have also been specifically addressed in the interconnected, autonomous database environment [HM85, LM84].

In this short paper, we briefly examine the **Remote–Exchange** project at USC. A major focus of this project is to devise and experimentally implement techniques and mechanisms to support the controlled sharing of information among a collection of autonomous, heterogeneous database systems. We shall not attempt here to examine the thrusts of this research project in detail, but rather briefly examine three of its principal distinguishing aspects.

First, the **Remote–Exchange** approach adopts a more general view of sharing among autonomous database systems than previously supported. In particular, we examine information sharing and coordination beyond the conceptual schema level; we note that sharing may be at a number of levels of abstraction and granularity, ranging from specific information units (data objects), to meta-data (structural schema specifications and semantic integrity constraints), to behavior (operations), to database model and supporting system (e.g., database management system). Our choice of an object-based common data model is a key to accommodating this sharing. Second, objects from external database systems are manipulated transparently by user and application level processes of the local database system. Here, integration of the *remote* objects into the local database system plays a vital role, as does the ability to compare objects at various levels of granularity and abstraction. Third, we employ the notion of an "intelligent" advisor to assist non-expert database system users in establishing and refining sharing patterns with external database systems.

2 Spectrum of Heterogeneity

As a basis for our analysis of and approach to the various kinds of diversity that may exist in the interconnected autonomous database environment, consider a federation of components, which are individual data/knowledge base systems. A component includes both structural and behavioral information. Structural information includes information units (viz., objects) and their inter-relationships, at various levels of abstraction; these represent both specific data facts and higher-level meta-data specifications. Behavioral information includes operations to manipulate information units and their inter-relationships; these operations can be invoked either by users or by the system itself; also included are services that a component may provide to itself or other components in the federation.

In this context, we can consider a spectrum of heterogeneity. That is, the heterogeneity in the federation may be at various levels of abstraction:

- **Meta–data language (conceptual database model):** The components may use different collections of and techniques for combining the structures, constraints, and operations used to describe data.
- **Meta–data specification (conceptual schema):** While the components share a common meta-data language (conceptual database model), they may have independent specifications of their data (varied conceptual schemas).
- **Object comparability (database):** The components may agree upon a conceptual schema, or more generally, agree upon common subparts of their schemas; however, there may be differences in the manner in which information facts are represented[Ken89]. This variety of heterogeneity also relates how information objects are identified, and to the interpretation of atomic data values as denotations of information modeled in a database (naming).
- **Low–level data form format:** While the components agree at the model, schema, and object comparability levels, they may utilize different low-level representation techniques for atomic data values (e.g., units of measure or description).
- **Tool (database management system):** The components may utilize different tools to manage and provide an interface to their data. This kind of heterogeneity may exist with or without the varieties described immediately above.

3 Accommodating Heterogeneity in a Federation

A cornerstone of the **Remote–Exchange** approach is the capability to incorporate remote objects directly into a component's local database. This allows users to use the local data manipulation facilities to access remote objects, and to combine local and remote objects. This results in substantial location transparency[1] .

Operationally, we view the steps in establishing sharing between components as first, the *discovery* of relevant information from remote components for importation into the local database. Having found these remote objects, the next step is to *integrate* them with the local component's objects. Both these processes will need to compare objects at different levels of granularity an abstraction, thereby requiring some notion of *object equivalence*. Finally, remote objects along with local objects are manipulated using the component's local data manipulation language/mechanism. Below we briefly explain specific issues involved in this framework and the **Remote–Exchange** mechanisms which address these issues.

[1] In some sense this can be viewed as an aspect of data independence.

3.1 Object–Based Common Data Model

In order for any sharing to take place among the heterogeneous components, some common model for describing the shared data must be established[2]. This model must be *semantically expressive* enough to capture the intended meanings of conceptual schemas which may have been designed from different perspectives using different data models. To this end we have chosen a Kernel Object Data Model (KODM) as the common data model for describing the structure, constraints and operations on the shared data.

At the same time, due to autonomy issues, a component may wish to exercise control over information that it exports by isolating the sharing to a fixed set of operations. This situation can also occur when the component does not support a general purpose data manipulation language/mechanism (e.g., a file system). KODM addresses this problem by allowing components to *encapsulate* the functionality of the shared objects.

Another benefit of using an object-based common data model is that it is *extensible*. In an environment where dynamic changes in component sharing patterns occurs naturally over time, this capability is indispensable. It allows components to gracefully adapt to new and unplanned operations on shared data.

A final aspect of KODM that we exploit in this project is *object uniformity*. Meta-data, the specific data, and operations are represented uniformly as objects. This allows the unification of various concepts(e.g., object equivalence).

3.2 Remote Database Transparency

Traditionally, DBMSs have been built as centralized repositories of information. Multiple users access this information in a uniform, controlled manner using tools provided by the DBMS. With the advent of networks, this basic architecture has been extended to what is called the client/server model. In this model, users (the clients) are now physically distributed among different machines. Yet, the fundamental asymmetry in the client/server relationship remains. The data is all stored and managed centrally. The servers also dictate the interface users must use for accessing and manipulating the data.

In the environment we are considering, the users may once again be physically distributed, however, they also maintain their own local database managed by their own local DBMS. These local databases may be very large or quite small as in a personal database. This results in two major issues:

1. The data is no longer managed by a single centralized agent.

2. Each local DBMS may have its own interface for accessing and manipulating data.

In this environment, our approach allows the incorporation of objects in remote databases into the user's local database. In so doing, remote objects should appear transparent to users of the local database. This means that the user manipulates the remote data the same way s/he manipulates the local data. In order to achieve this transparency, we exploit the *encapsulation* and *object uniformity* features of KODM to form a generalized framework for the execution of (remote) operations[Sha89].

3.3 Discovery

The preceding discussion has in a sense assumed that the objects being shared were already known. The *discovery* process pertains to finding out what information should be shared in the first place. At the most abstract level, the remote objects that would probably have most interest to the user of a local database are those objects whose concept domain overlaps with the concepts of his/her local database. Depending upon the type of information which a user is interested, s/he may wish for a large intersection or a small one. Large intersections would correspond to concepts that are very similar to the concepts in his/her database (e.g. Person and Employee). Small intersections would correspond more to "related" type of information (e.g., House and Owner), or completely disjoint information.

To assist users in discovering, establishing and refining these sharing patterns with external database systems, we explore the notion of an *intelligent* **Sharing Advisor**. In order to provide this functionality, the **Sharing Advisor** must have access to information which describes (meta)data and possible relationships among them in the various components. This information is stored in the **Semantic Dictionary** which is simply another component in the federation, accessed by the **Sharing Advisor** (and possibly other components).

[2] Another alternative would be to deal with the heterogeneity on a component pairwise basis which would require n^2 translators. We feel that the scaling problems associated with the translators precludes this approach.

In particular, the sharing advisor exploits the following techniques:

- *Structural and behavioral equivalence*: By using structural and behavioral equivalent techniques, we can identify semantically related objects. This applies to similarities among exported objects and similarities between exported objects and the requested information (see next section).
- *Constraint analysis*: Constraint analysis may be utilized to identify semantically related objects; the knowledge of constraints among remote objects suggests their potential relevance and relationships with local objects[UD88].
- *Probabilistic approach*: This approach has been very active in the area of information retrieval research. The basic idea is to assign a probability index, with respect to a query, to each object which reflects the probability that the object is relevant to the requested information [Fuh90].
- *User feedback data*: For each retrieved information, a feedback data is returned from the user based on how relevant the retrieved information is with respect to the query [Fuh90]. The feedback data should then be integrated into the upcoming retrieval process to enhance the accuracy of the probability estimate.
- *Heuristics*: We can consider the initial set of heuristics to be some information/rules used to help in identifying semantically similar objects and in estimating the initial probability relevance of each object. On the other hand, heuristics will be dynamically incremented from time to time due to the feedback data from users which may enhance the accuracy of the probability estimation process.
- *Machine learning techniques*: By applying machine learning techniques based on heuristics and user feedback data, the advisor will identify semantically similar objects more accurately, and thus retrieve more relevant information. In addition, machine learning techniques can also use to dynamically update the semantics of each object whenever the object in the corresponding component database system is altered [LM88]. In this sense, it helps in establishing semantic knowledge of exported objects which acts as a basis for determining object similarities.

3.4 Object Equivalence

One of the fundamental goals of **Remote–Exchange** is to automate the sharing of information objects among a collection of database components. In general, sharing is possible at many different levels of abstraction and granularity, ranging from specific information units (data objects), to meta–data (structural schema specifications and semantic integrity constraints), to behavior (methods/operations). In light of such diversity, the ability to compare these various objects is required. Most of the previous work in this are has concentrated in the area of *structural equivalence* at the schema level. In addition to this, we are also exploring *behavioral equivalence* mechanisms. Determining whether objects are equivalent based on their behavior appears quite promising, as it allows for comparisons at different levels of abstraction.

3.5 Integration/Resolution

When a component wishes to import information from other participants of the federation, it first consults with the sharing advisor to establish a sharing pattern and locate possible sources of information. Once the relevant data has been discovered, the local component may add the remote (meta)data to its own local schema through the integration process. Adding (meta)data to an already existing schema is a two–step process. First, all conflicts (e.g., naming, structural, scaling) between the local database objects and the external objects must be resolved. Second, these objects must be integrated into the local schema as gracefully as possible. Since the objects being imported can represent data-values, meta-data, or operations, this task can be difficult. To help in this process, the integration process needs to access information in the **Semantic Dictionary** described earlier for information about external objects.

4 Conclusions and Research Directions

This short paper has examined the problem of supporting dynamic patterns of sharing among a collection of autonomous, heterogeneous databases. We have explored the spectrum of heterogeneity that may exist in this environment, and have described a top level architecture and approach to sharing. The experimental **Remote–Exchange** system under development explores a number of architectural and implementation issues, as well as providing a basis for the refinement of the sharing model and mechanism.

The principal current focus of research centers on the following issues:

- Detailed design and implementation of the remote-transparency mechanism in particular, the transparent sharing of behavioral objects among components;
- Efficient (remote) execution of these behavioral objects;
- Refinement of the kernel object database model, and experience with the use of the extensible features of the model;
- The content and organization of the **Semantic Dictionary**;
- The collection of heuristics that form the basis for the operation of the sharing advisor;
- Studies of the tradeoffs involved in centralized vs. decentralized control and management of coordination information;
- Access control (a related project is currently examining this);
- Utilization in specific application domains (viz., collaborative design environments and computer-integrated manufacturing); and
- Employing **Remote–Exchange** as a framework for the interconnection of heterogeneous database management systems (O2, Ontos, Orion, IRIS, and a relational system are being examined in this context).

References

[BLN86] C. Batini, M. Lenzerini, and S. Navathe. A Comparative Analysis of Methodologies of Database Schema Integration. *ACM Computing Surveys*, 18(4):323–364, 1986.

[CP84] S. Ceri and G. Pelagatti. *Distributed Databases: Principles and Systems*. McGraw Hill, 1984.

[DH84] U. Dayal and H. Hwang. View Definition and Generalization for Database Integration in Multibase: A System for Heterogeneous Distributed Databases. *IEEE Transactions on Software Engineering*, 10(6):628–644, 1984.

[Fuh90] Nobert Fuhr. A probabilistic framework for vague queries and imprecise information in databases. In *Proceedings of the International Conference on Very Large Databases*, pages 696–707, 1990.

[HM85] D. Heimbigner and D. McLeod. A Federated Architecture for Information Systems. *ACM Transactions on Office Information Systems*, 3(3):253–278, July 1985.

[Ken89] W. Kent. The Many Forms of A Single Fact. In *Proceedings of the IEEE Spring Compcon*. IEEE, February 1989.

[LA86] W. Litwin and A. Abdellatif. Multidatabase Interoperability. *IEEE Computer*, 19(12):10–18, December 1986.

[LM84] P. Lyngbaek and D. McLeod. Object Management in Distributed Information Systems. *ACM Transactions on Office Information Systems*, 2(2):96–122, April 1984.

[LM88] Q. Li and D. McLeod. Object Flavor Evolution in an Object-Oriented Database System. In *Proceedings of the Conference on Office Information System*. ACM, March 1988.

[MB81] A. Motro and P. Buneman. Constructing Superviews. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, Ann Arbor, Mich., April 1981. ACM SIGMOD.

[Sha89] M. Shan. Unified Access in a Heterogenous Information Environment. *IEEE Office Knowledge Engineering*, 3(2):35–42, August 1989.

[UD88] S.D. Urban and M.L. Delcambre. Constraint analysis: A tool for explaining the semantics of complex objects. In *Advances in Object-Oriented Database Systems*, pages 156–161. Springer-Verlag, 1988.