SOME THOUGHTS ON DATA SHARING AMONG AUTONOMOUS COOPERATING DATABASE SYSTEMS

Rafael Alonso
Hector Garcia-Molina

CS-TR-100-87

June 1987

SOME THOUGHTS ON DATA SHARING AMONG AUTONOMOUS COOPERATING DATABASE SYSTEMS

Rafael Alonso
Hector Garcia-Molina

Department of Computer Science
Princeton University
Princeton, N.J. 08544
(609) 452-3869

ABSTRACT

As our society becomes increasingly information dependent, there will be ever greater pressures for information sharing among separate entities. Organizations will find that, in order to cooperate efficiently, they will have to share data with their partners. The rapid spread of networking technology will speed this process along, since it will become more convenient, faster, and less expensive than ever to communicate with others. However, although the benefits of cooperation are self-evident, it seems clear also that many organizations will not be willing to surrender autonomy over their data as the price of cooperation

In this paper we describe our current thoughts about the issues involved in sharing information effectively among a very large number of independent database systems. A major point of our work is to preserve the autonomy of the participants while still allowing efficient sharing. This work has as a starting point the concepts behind *federated databases*, but we are expanding these ideas to apply to systems with much larger numbers of participants than usually considered, supporting very high transaction rates, and allowing for a greater degree of heterogeneity than in previous research.

July 13, 1987

SOME THOUGHTS ON DATA SHARING AMONG AUTONOMOUS COOPERATING DATABASE SYSTEMS

Rafael Alonso
Hector Garcia-Molina

Department of Computer Science
Princeton University
Princeton, N.J. 08544
(609) 452-3869

1. Introduction

As our society becomes increasingly information dependent, there will be ever greater pressures for information sharing among separate entities. Organizations will find that, in order to cooperate efficiently, they will have to share data with their partners; individuals will be interested in obtaining information and businesses will provide it for a fee (as Dow Jones [Dunn1984] and the Source [Edelhart1983] do today). The rapid spread of networking technology will speed this process along, since it becomes more convenient, faster, and less expensive than ever to communicate with others. However, although the benefits of cooperation are self-evident, it seems clear also that many organizations will not be willing to surrender autonomy over their data as the price of cooperation.

We are currently investigating some of the issues involved in sharing information among a large number of independent users. By sharing we mean applications that perform both reading and updating operations. The term sharing also implies a kind of symmetry; i.e., there are no ``providers'' and ``consumers'' of data, but rather users will be cooperating by exchanging data. And they will not only be interested in someone else's information but also in integrating foreign data with their own in an efficient manner.

The information we have in mind will not be limited to the usual bank data; instead, it will be represented in heterogeneous ways (i.e., graphical data, telephone conversations, television shows), each having different transmission requirements. The data items may also have certain relationships among them (i.e., a set of video images is associated with a soundtrack), which may have to be preserved across autonomy domains. The information may also consist of incomplete or partially incorrect data. This condition may arise on purpose out of the data owner's desire to maintain certain information private even though sharing data. For example, an organization may share its employee file with a health-insurance provider, but the latter has no need to access salary information, so that part of the information may be deleted or randomly altered.

We expect that there will be large numbers of organizations involved in the activities described above. Since each owner of a personal computer could be classified as a cooperating entity, the number of participants in a sharing scheme could be in the millions. The main impact of the size of the overall system will be that whatever schemes for cooperation we devise will have to scale to serve a possibly global system.

The salient feature of our current work is the emphasis on the autonomy of the cooperating entities. Since we assume that the participants will belong to different organizations and have (potentially) conflicting requirements, we cannot expect that a sharing strategy will be successful if it forces anyone to unwillingly service another party. Furthermore, participants may at any time remove themselves from the network and cease to provide whatever information they were offering previously.

It seems clear that there is a distinct tension between sharing and autonomy, at least for given levels of transparency and efficiency. For example, consider a parts supplier that would like to allow an important client to look at the supplier's inventory database and modify entries in it (for example, to decrease the number of parts available if the manufacturer places a large order for them). While this cooperation may be of benefit to both parties, the supplier may be unwilling to agree to it because his client may lock frequently used data during very long transactions; or perhaps the supplier does not want to allow the client access during certain busy times of the year. However, if the manufacturer is willing to forego transparency (in this context, the ability to think that the parts information is just like any other data that belongs to his own database) the sharing may still be possible; the manufacturer will just have to be aware of the fact that some data is located elsewhere and may sometimes be inaccessible. Alternatively, if the manufacturer has a similar relationship with other suppliers there is another approach that maintains transparency (albeit at the cost of decreased efficiency). Instead of assuming that a supplier's database is always available, the manufacturer's database software could use a simple protocol whenever it contacts a supplier database to determine if cooperation is possible. If that particular database is inaccessible, the software can then try another supplier and the manufacturer will not have to concern himself with the actual location of the parts information.

In light of the requirement for autonomy, some of the techniques normally considered for sharing are not appropriate. For example, moving all the information to a central database is clearly not the answer, since in this approach users may lose all control over their data. A better idea might be to preserve the physical distribution of the data but to integrate all the cooperating databases into a single distributed database (such as R* [Williams1982] or distributed INGRES [Stonebraker1977]. This would ensure efficient sharing as well as preserving a large measure of transparency. However, there is some autonomy lost. In typical distributed database systems, a site wishing to access remote data will be allowed to lock that information and keep other transactions waiting. Thus, the owner of the data might not be able to access it for a very long time. Similarly, in some distributed commit algorithms, there is a coordinating site which tells other sites whether to abort or commit. If some remote site loses its communications with the coordinating site, it will either have to wait until communications are restored or risk creating a database inconsistency. (See [Lindsay1980] for a more thorough discussion of autonomy issues in distributed databases.)

A still better approach is that of creating a federated database [Heimbigner1985]. In such a system, the autonomy of the individual databases is quite important, and they are free to join or leave the federation at will. A federation maintains a *federal dictionary* which contains information about the formation of the federation itself. Individual sites describe what they are willing to share via *export* schemas, and they have *import* schemas that refer to the remote information that they can access. In [Heimbigner1985] a mechanism is described by which participating sites can negotiate the type of information they will share, as well as the kind of access to be granted.

We feel that the ideas present in federated databases serve as an appropriate starting point for the class of environments we are studying. However, there are a number of issues that require further research. For example, when the number of participants is extremely large, the notion of a single federal dictionary may break down. As Heimbigner and McLeod point out, some of the strategies employed in distributed name servers [Oppen1983, Birrell1982] may be required. But since we expect that a single database may belong to many federations, a higher level mechanism may be required (at its simplest, a private directory that specifies which name server to query for a given item of information). An even more difficult matter to resolve is that of heterogeneity. Previous work on federated databases has dealt with the case where all the database systems are identical. But unless organizations are aware of the need to cooperate before designing their local systems, the databases that they will use will be of different types and will structure data in varied fashion. Some of the strategies employed in heterogeneous databases [Landers1982] may be of help in the current context. Another class of questions arise when one becomes interested in implementing in an efficient manner the functionality we require. That is, we are also concerned with supporting a relatively high interaction rate among the participants, as well a minimizing the overhead involved in sharing. These and other problems are described in greater detail in the following section.

2. Research Problems

In this section we address some of the issues that we feel are involved in understanding sharing among autonomous databases. It should be understood that this is by no means an exhaustive list of all the problems in this area, only an outline of the more salient ones. We have already done preliminary work on some of these questions, and the rest will be tackled in the near future.

2.1. Finding the Information

As we have pointed out, a single database dictionary may not be sufficient to allow users to find the data they wish. This is true partly due to the large amount of information it would have to contain, but also because the cooperating sites might be located far apart, thus requiring a distributed solution (e.g., a name server). But the mechanism required will be more complex than a name server. For example, there will be systems involved in partnerships that will not want others to know what information is being shared (or even that there is any sharing taking place). Furthermore, since we expect that databases will belong to many federations, each system will have to contain a complex data structure that will associate information items with the name server to be used in locating them.

A related decision involves specifying the level at which information will appear in these naming schemes. That is, if an organization is interested in sharing employee data with another, it is not clear whether the naming service should contain an entries for each employee, each department, or a single entry for the company (i.e., for the employee relation in the database). Clearly, the answer to this question depends on the level at which sharing is done by the databases, but also on the amount of information the naming scheme can handle efficiently.

For truly large systems, even a complex naming service might not be sufficient. For example, in many cases, a user will find that no one has registered the location of an item with the service, simply because they were unaware of anyone else possibly requiring it. Or perhaps a user knows that many databases have information about a topic but he is interested in a more specific sub-topic. In such cases, one possible solution might be to send out a query that will travel to associated machines trying to determine if anyone has the information. However, such *roving queries* will have to be carefully designed, since it would be easy to destroy the usefulness of the union if they are too frequent or too time-consuming to process. To limit their number, groups of roving queries from all participants could be batched and sent at periodic interval (similarly to the way want ads are now handled by newspapers). In order to speed up their processing, roving queries could be limited to a simple keyword match that either succeeds or fails. After all, their purpose is not necessarily to get the information, but rather to determine where it lies.

Finally, since participants are free to modify their local databases and not register the changes until it is convenient to them, it is likely that the information on data placement will become outdated for some items. If so, this information will have to be treated as if it were no more than a hint as to the location of the data.

2.2. Caching

The use of caching techniques seems attractive in order to implement sharing more efficiently. Caching should decrease traffic on the network and improve the response time of queries. We have already done some simulation studies of caching over phone lines in the context of public information systems [Simpson1987]. We now intend to extend this work to include faster communication services as well as two-way communication exchanges.

However, one of the problems of using the ''standard'' caching studied in [Simpson1987] as well as in other studies, is that, in the case of information updates, the data owner must invalidate all caches. Clearly, to maintain cache coherency imposes an obligation on the data owner. On the other hand, if cached data is allowed to diverge from the original, the usefulness of the cached image would be lost, since users would have no idea of how different the image is from the actual value.

A promising compromise is to allow caches to become incoherent, but in a controlled fashion. For example, let us assume that a user is satisfied as long as his cached data is within 10% of the actual value. If so, the site that owns the data will not have to send invalidation messages (or the new values), as long as that condition is met. A second condition might be that the cache image is guaranteed to be within 10% of the actual value but only for 24 hours. Thus, the owning site knows that when that time expires its responsibility towards a sharing site will end.

We call data copies that are allowed to diverge in a controlled fashion *quasi-copies*. In [Garcia-Molina1986a] we describe a variety of policies for such entities, and discuss implementation ideas. It is worth stressing that quasi-copies are particularly effective for autonomous cooperation not only because they have the potential advantages of caching (i.e., less network traffic and improved response), but because they provide a ready framework for specifying an owner's degree of responsibility towards those who share his data. Finally, in [Garcia-Molina1986a] we also describe how a quasi-copy implementation can be adapted to cope relatively well with network partitions.

2.3. Concurrency Control

A possible model for database interaction consists of utilizing what we have called *global procedures* [Alonso1987a] Such procedures are used in the following manner. When a user wants to make a request involving a group of cooperating databases, he executes a global procedure. Such a procedure spawns a local transaction at each of the databases on the user's behalf. The local transactions are written in the query language of each particular database, and in terms of the local schema. All the local transactions are independent of those at other sites; as far as each database is concerned, these transactions are no different from those invoked on the behalf of local users. There is a process at each database site to coordinate the execution of the global procedure.

There are many details of global transactions that we need to resolve (e.g., the manner in which global procedures are decomposed into local transactions). For the present we have concentrated on concurrency control mechanisms. The most obvious choice for serializing global procedures (i.e., two-phase locking) has the following problem in the current context. If one of the local transactions invoked as part of a given global procedure takes a long time to complete, all other global procedures that may conflict with the stalled one will have to wait. Since the potential wait may be large (after all there may be many cooperating databases located very far apart) we have developed two concurrency control mechanisms, *sagas* and *altruistic locking*, which are suitable for very long lived interactions.

Altruistic locking is an extension of two-phase locking which also guarantees serializability. The essential idea is that long lived transactions will release early the locks on data that they will no longer access. The details of the strategy, as well as a further extension, are provided in [Salem1987].

For many applications it is not necessary to treat global procedures as transactions. A *saga* [Garcia-Molina1987b] is a procedure that can be broken up into a collection of smaller transactions which can be interleaved in any way with other transactions. The transactions in a saga are related to each other and should be executed as a (non-atomic) unit. Since global procedures for cooperating databases are collections of service requests (i.e., local transactions), they are natural as sagas if the application semantics do not require that they be executed atomically.

As a final comment, it should be pointed out that there is no need to treat the local transactions spawned by a global procedure in the same manner as normal (i.e., strictly local) transactions. It may be appropriate to treat them as lower priority transactions in a variety of ways: lower scheduling priority, choosing them for rollback if a deadlock occurs, aborting them if they have locked data that a strictly local transaction requires, etc.

2.4. Heterogeneity

As we mentioned in the introductory section, we expect that sometimes two entities with different database management systems and dissimilar data schemas will wish to cooperate. We also stated that the research carried out on heterogeneous databases would be of help in this work. However, most previous work in this area requires building a very complex global schema and developing sophisticated translation packages. This may be appropriate for long term cooperation, but may prove too costly for short term interactions. It seems to us that a promising approach is to make use of an expert system that will initially have enough knowledge to translate only some simple queries, and as the needs of the interaction grow, more information can be added to the system to facilitate more complex requests.

We have described in more detail the architecture of such a system in [Alonso1987b]. Our current design calls for an expert system written in Prolog to bridge a DBASE database (on an IBM PC) and an INGRES [Stonebraker1976] database (running on a DEC 11/750), both connected via an Ethernet [Metcalfe1976]. Although a complete prototype is not yet running, work has started, and two independent study projects have been proceeding for the past few months.

2.5. Other Issues

There are many other problem areas that require exploration. For example, we are not sure of what kind of networking environment our proposed solutions will require. It is clear that a high bandwidth will be required, but it is less clear what latency and failure modes we will tolerate.

How the databases will negotiate for cooperation is a difficult problem. It seems to us that in some instances a single complex negotiation protocol such as that implemented in [Heimbigner1985] may be fine. However, it seems reasonable that the type of cooperations that may arise in the future will not be obvious *a priori*, and any such scheme for implementing cooperation will fall short for many applications. Perhaps an expert system approach like the one described in the previous section will be more flexible and easier to extend.

It is not yet evident to us what level of standardization we can expect to occur in the real world. The more heterogeneity that exists the harder our task will be, and we expect that a large degree of variety will be present. However, market pressures may come to bear upon database administrators and perhaps there will be a large degree of standardization. It seems to us probable that, in order to ensure efficient cooperation, users will be willing to standardize where it is convenient to do so and go their own way in other areas; our research should take this point into account.

3. Conclusions and Future Work

We believe that data sharing among a large number of autonomous, cooperating database systems raises many challenging problems. Solutions to these problems will lead to the next generation information and data management systems. Since we do not have a good understanding of all the issues, our current studies are exploratory in nature. The main focus is the generation of new ideas for such an environment, as well as an evaluation of their limitations and capabilities.

At this early stage of research, it would seem difficult to build or even envision a complete prototype system. On the other hand, we feel it is very important to experimentally evaluate some of the more promising algorithms or ideas. Thus, in addition to using analysis and simulation, we plan to build a simple testbed system where components or particular algorithms can be evaluated.

Our testbed will be built on the facilities of our Distributed Computing Laboratory, which consists of a varied mix of machines (Sun workstations, Vaxes, IBM PC's, NCR

Towers, AT&T 3B2's, etc) connected by a stand-alone Ethernet. At first, we will experiment with identical (i.e., compatible data schemas) autonomous INGRES databases. We will then add a measure of heterogeneity by considering INGRES databases with incompatible schemas, and finally, different databases running on different hardware.

Another crucial aspect of our research plan is close cooperation with interested manufacturers. From our point of view, we need to obtain information on particular applications, traces of user requests, details on hardware configurations, and as much feedback as possible from our industrial partners. This will ensure that our solutions are realistic and meaningful. From the point of view of our industrial affiliates, they will receive our research results directly, via technical reports and mutual visits.

As a final comment, we should add that the coming year should be an especially active one at Princeton in the area of databases. Apart from the two principal investigators, we will be joined by an associate researcher, Daniel Barbara, and a visiting professor, Nancy Griffeth (from Georgia Tech), both of which are in the area of databases and plan to work with us. We expect that there will about ten graduate students and five undergraduates also involved in database-related projects.

References

Alonso1987a.
Alonso, Rafael, Hector Garcia-Molina, and Kenneth Salem, ''Concurrency Controls for Global Procedures in Federated Database Systems,'' Technical Report CS-TR-099-87, Department of Computer Science, Princeton University, 1987.

Alonso1987b.
Alonso, Rafael, ''A Prototype for Research on Heterogeneous Database Systems,'' *7th Conference of the Chilean Computer Science Society*, 1987.

Birrell1982.
Birrell, A., R. Levin, R. M. Needham, and M. D. Schroeder, ''Grapevine: An Exercise in Distributed Computing,'' *Communications of the ACM*, vol. 25, no. 4, pp. 260-274, April 1982.

Dunn1984.
Dunn, Bill, ''Bill Dunn of Dow Jones: The Data Merchant,'' *Personal Computing*, pp. 162-176, December 1984.

Edelhart1983.
Edelhart, Mike and Owen Davies, *OMNI Online Database Dictionary,* Collier MacMillan Publishers, 1983.

Garcia-Molina1986a.
Garcia-Molina, Hector, Rafael Alonso, Daniel Barbara, and Soraya Abad, ''Data Caching in an Information Retrieval System,'' Technical Report CS-TR-065-86, Department of Computer Science, Princeton University, 1986.

Garcia-Molina1987b.
Garcia-Molina, Hector and Kenneth Salem, ''Sagas,'' *Proc. ACM SIGMOD Annual Conference*, pp. 249-259, San Francisco, CA, May, 1987.

Heimbigner1985.
Heimbigner, Dennis and Dennis McLeod, ''A Federated Architecture for Information Management,'' *ACM Transactions on Office Information Systems*, vol. 3, no. 3, pp. 253-278, July 1985.

Landers1982.
Landers, T. A. and R. L. Rosenberg, ''An Overview of Multibase,'' in *Distributed Databases*, ed. H. J. Schneider, North-Holland, 1982.

Lindsay1980.
Lindsay, Bruce and Patricia G. Selinger, ''Site Autonomy Issues in R*: A Distributed Database Management System,'' Research Report RJ2927 (36822), IBM San Jose Research Laboratory, September 15, 1980.

Metcalfe1976.
Metcalfe, R. M. and D. R. Boggs, ''Ethernet: Distributed Packet Switching for Local Computer Networks,'' *CACM*, vol. 19,7, pp. 395-404, July 1976.

Oppen1983.
Oppen, D. C. and Y. K. Dalal, ''The Clearinghouse: A Decentralized Agent for Locating Named Objects in a Distributed Environment,'' *ACM Transactions on Office Information Systems*, vol. 1, no. 3, pp. 230-253, July 1983.

Salem1987.

Salem, Kenneth, Hector Garcia-Molina, and Rafael Alonso, ''Altruistic Locking: A Strategy For Coping with Long Lived Transactions,'' *Second International Workshop on High Performance Transaction Systems*, 1987.

Simpson1987.

Simpson, Patricia and Rafael Alonso, ''Data Caching in Information Retrieval Systems,'' *Intl. Conference on Research and Development in Information Retrieval*, 1987.

Stonebraker1976.

Stonebraker, M. R., E. Wong, P. Kreps, and G. D. Held, ''Design and Implementation of INGRES,'' *ACM Trans. on Database Systems*, vol. 1, no. 3, September 1976.

Stonebraker1977.

Stonebraker, M. R., ''A Distributed Version of INGRES,'' *Berkeley Workshop on Distributed Data Management*, Lawrence Berkeley Laboratory, May 1977.

Williams1982.

Williams, R. et al, ''R*: An Overview of the Architecture,'' *Proc. of The International Conf. on Data Bases*, Israel, June 1982.