The Issues And Solutions Of Integrating DBMS To A Multi-DBMS

Leslie Leong, (E-mail: LeongL@ccsu.edu), Central Connecticut State University

Abstract

Many organizations invest heavily in heterogeneous databases according to organizational functions. These heterogeneous databases are stand-alone systems that do not interact with one another. The objective of this paper is to introduce a multi-database system (MDBMS) that interacts with other heterogeneous DBMS within the organization to integrate information processing. In this paper, we discuss the potential inconsistencies in integrating heterogeneous databases. We further extend to include issues in designing a MDBMS. With a MDBMS, data sharing across organization reduces overheads and costs, thus, provides a competitive advantage to the global firms.

1. Introduction

Information is a crucial resource in business' daily operation. In the past, information is usually shared by utilizing telephone, facsimile, videotape, audiotape, as well as documents. Today, a lot of them have been replaced by information technologies. Voice mail facilitates the information exchange. Electronic mail improves communication. Even archives are transferred electronically. The traditional business documents were paper documents. To date, electronic documents have replaced paper documents in many businesses. For the information technologies grow rapidly, we can foresee that there must be more information technologies to be utilized in business operations.

Palmer et at. (1995), stated the importance of information management that "The very essence of an organization is information. When recorded, information becomes a very real and tangible element in the operation and management of organizations - private or public, profit or nonprofit, and large or small. Information has no usefulness, however, unless it can be retrieved, manipulated (forted out or combined with other information), and placed at the disposal of those who need it" (p.170). Computerized environment provides more efficient and effective access for the employees to manage and/or retrieve the information.

Data or information is stored in a database. Litwin (1993) defined a database as "a centrally defined collection of data" (p.14). Moreover, a database is a collection of data, and retrievable information. There are many advantages to centralize related data and put them under one single control. Centralized data allows users to find relevant information easier and more efficient. Also, the management of data will be more effective. Users access data through the help of the database management systems (DBMS), which have full control of the data stored in the repository. DBMS provide essential functions as well as user-database interfaces for users to add, retrieve, modify, or erase data. (Klaus et al. 2000; Seung et al. 2001).

All too often, enterprise information systems consist of a diverse mix of applications, files, and databases that are each individually essential but do not cohere well as a whole (Singh et al., 1997). Over time, databases in different departments and environments tend to be developed individually to meet their requirements. Moreover, variant DBMS are usually not compatible with each other. However, as the aforementioned discussion, sophisticated user requires information integrated from several sources. A single, individual, and isolated database no longer satisfies user's demand nowadays. Consequently, being able to manage several heterogeneous databases becomes an issue. Heterogeneous databases are databases that have different structures, data types, stored formats, or interpretations for the data. Due to the variant natures of the heterogeneous databases, it is not possible to manage all of them

Readers with comments or questions are encouraged to contact the author via email.

with a single database management system. It is also unreasonable and impossible to transform enormous data into one single, unique format. Therefore, the question to allow global users to access data they need between heterogeneous databases is a problem needed to be solved.

Based on the aforementioned introduction, this paper present a method of managing heterogeneous databases. Many issues with regard to this topic have been discussed and numerous studies have been conducted. This paper synthesized some of those works and presented a means for organizations to converge their heterogeneous databases with the least effort.

Let us take a look at this scenario. The XYZ Company has a large number of heterogeneous computer systems that have critical data for decision-making and customer growth. The company's databases range from spreadsheets in stand-alone PCs to Mainframe VTOC databases. The data is currently not shared among the systems, and tedious expensive work is necessary to consolidate the huge amounts of data, which includes duplication and copying of databases across different computers.

The goal of this paper is to converge the aforementioned databases without modifying them. This includes the databases themselves as well as the DBMS. The issues reside in both the databases and the DBMS. However, the goal is to develop a system or a method so that data interchange will occur on the conceptual level above the DBMS. Thus, the DBMS need only minimum or no change. From the user's point of view, they will not know where the information comes from and how differently it is stored.

Bright et al. (1993) proposed a strategy stating that "A multitude of systems usually means multiple access methods and user paradigms. It is unreasonable to ask users to learn all these access methods, yet it is also unreasonable to expect organizations to convert all their systems to a single common data model with a single access method. Multidatabase systems give users a common interface to multiple databases, while minimizing the impact on existing database operations" (p.3). This paper is going to take the idea and develop a multidatabase systems in order to integrate these local databases.

2. Literature Review

Palmer et al. (1995), defined heterogeneity of hardware as "the presence of multiple hardware platforms and reflects the shift from customers relying on just one supplier to buying from many" (p.81). The same concept can be referred to software as well as other information facilities.

The heterogeneity of databases can be derived from many sources. One source is the nature of the data. Variant data has its unique way of storing. For instance, texts, graphs, images, spreadsheets, tables, and animation all have different means of storing. Some documents themselves consist of more than one type of data. HTML document is a representative of this kind of complex document that consolidate text, sound, image, or full-motion video.

Another example of the heterogeneity is the format of storing data. Texts can be stored in ASCII form in PC or EBCDIC form in mainframe. Images have forms of bit-mapping, JPEG, GIF, and so on. Even the same spreadsheet can be stored in different forms by different application. The scenario is inevitable because the documents might be developed on variant platforms, strategies, or for variant uses. Enormous resources of gathering data can result in the same problem as well. The job of a DBMS is merely helping users to locate and access data. Thus, the issues cannot be solved in DBMS level.

Another main reason which causes heterogeneity is the structure of the database. Since the first database is built, there have been four main categories of databases. They are hierarchical database, network database, relational database, as well as object-oriented database. Although some legacy systems remain using hierarchical or network databases, the most popular are relational databases nowadays. For the essences of these databases are extremely different, the data searching and translating are inefficient and ineffective.

The main issues of integrating heterogeneous databases have been discussed widely (Anne et al. 2000; Joon H.L. 1998). Many researches focused on discovering and analyzing those issues. With the clear understanding of the issues, we can then develop an effective solution.

Bouguettaya and Milliner (1995) have proposed that "the complexity of making autonomous heterogeneous databases smoothly interoperate is dependent on addressing two major issues. The first issue to address is what adequate levels of autonomy databases are guaranteed to keep. The second issue to address is what overhead cost is required to bridge database heterogeneity" (p.1388). The statement conveyed two messages. Firstly, we want to construct global communication environment for local databases while keeping the databases as fully autonomous as possible. Secondly, we want to reduce cost. Both of these concerns point to the solution to one direction which is to build a multidatabase system (MDBMS). The MDBMS manages the local databases so that the local databases retain their autonomy, thus reducing overhead time and effort on restructuring every local database.

Bright et al. (1993a) proposed their analysis of the issues of managing heterogeneous databases. There are several issues that cause the integration of heterogeneous databases a challenging job to achieve. Three of them are most fundamental but crucial and will be discussed in this paper. They are *Site Autonomy*, *Differences in Data Representation*, and *Concurrency Control*.

2.1 Site Autonomy

Bright et al. (1993a) defined site autonomy as "A key aspect of multidatabases, as opposed to distributed databases, in that each local database management system (DBMS) retains complete control over local data and processing. The DBMS itself is not modified by joining the multidatabase" (1993b, p.5). According to this definition, a DBMS of an autonomous site have full control to access, manipulate, and modify the data stored in its local databases. Although the purpose of multidatabase is to allow global databases to access data stored in different local databases, those global databases are not supposed to have direct access to the local databases. They are supposed to interact with the local DBMS instead of retrieving data directly from local databases. Since the global system interfaces with the local DBMS at the user level, the local DBMS sees the global system as just another local user (Bright et al. 1993a).

The importance of site autonomy was described as "local databases that have critical role in an organization, and it may be impossible from an economic standpoint to change these systems. Site autonomy means the local DBMS can add global access without changing this existing local function" (Bright et al. 1993b, p.5). Without autonomy, a DBMS can seldom maintain the data integration of its database. There would have to be some additional procedures for the purpose of retaining the database's integration.

Site autonomy can also act as a security measure because the local DBMS has full control over who accesses local resources through the multidatabase interface and what processing options will be allowed (Bright et al. 1993 p.5). Hence, the DBMS is able to prevent unauthorized invader and improper modifications.

2.2 Different data representation

Bright et al. (1993a) indicated that different data representation is inevitable in a multidatabase environment. "Because local databases are developed independently with differing local requirements, a multidatabase system is likely to have many different models, or representations, for similar objects" (p.6). The problem of different data representation is that the recognition of data becomes complicated. Without the right recognition, the translation of the data is ineffective. Consequently, even the data retrieved is the right one, it is useless.

The differences of data representation are due to many causes. Bright et al. (1993a) categorized them into four types: *Name Differences*, *Format Differences*, *Structural Differences*, and *Missing or Conflicting Data*.

Name differences result from the local databases' different conventions for naming objects. There are two types of name differences, *Synonym* and *Homonym*. Bright et al. have given explicit ideas of them and the counter-

measures for the global systems. "Synonym means the same data item has different names in different databases. The global system must recognize the semantic equivalence of the items and map the differing local names to a single global name. Homonym means different data items have the same name in different databases. The global system must recognize the semantic difference between items and map the common names to different global names" (1993, p.6).

Format differences include differences in data type, domain, scale, precision, and item combinations (Bright et al. 1993a, p.6). It is derived from that local databases define data differently. A series of digital can be an amount of deposit in one database and social security number in another database. Typically, this problem can be solved by utilizing transformation functions between local databases and global representation. However, a further problem in this area is that the transformation from local to global may be easy but the inverse transformation may be complex.

Depending on the use of the data, an item may be stored in different structures in different databases. For instance, an item may be a value in one database and an attribute in another. Those structural differences result from their semantic definitions, which are often seen in multidatabases.

Missing data refers to those data that are not recorded completely due to incomplete updates, system errors, or power failures. Usually, the system can still retrieve the data from the database. However, the value of the data is no longer meaningful. The problem is that the system may not be able to perceive this kind of errors. Whether the data is well recorded or not, there will be a value stored in where the data should be stored. It might be the previous data. If the system is able to detect whether the data is out of range and the data is out of range due to the recording failure, the data can be discarded. If the system takes the wrong data as if it is right, due to the garbage-in-garbage-out rule, the process that uses the data will generate useless information.

Sometimes the same data stored in different databases may have conflicts at the same time frame, which means their values are variant and none of them is recorded later than the other. The differences may be due to several reasons. Whatever it is, the system can no longer locate the right data for processing. Consequently, either the system cannot provide qualitative service or the performance becomes ineffective.

2.3 Concurrency

Multidatabase management systems must be able to run two or more transactions simultaneously. This ability is known as transaction concurrency (Bobak, 1996). In a multidatabase environment, the global system and local DBMS may manipulate the same data simultaneously. If any side fails to finish its transaction without interrupting, the worst result will be that both systems get the invalid value. Thus, the multidatabase environment must find a way to control the concurrency.

"Related information vital to a global application or request may exist in multiple, incompatible local databases. Users cannot be expected to manage the system detail of sending multiple requests in different languages to multiple information sources (possibly different data models). Multidatabase systems provide integrated, global access to autonomous, heterogeneous local databases via a single, relatively simple request" (Bright et al. 1993b, p.1).

The necessity of an integrated system for multiple heterogeneous databases is essential because users cannot be expected to deal with this complex environment alone, as Bright et al. said. A multitude of solutions has been proposed to fulfill the goal. One strategy that has been discussed a lot and been proven to be effective is multidatabase.

Bright et al. (1993b) has defined that "A multidatabase is a distributed system that acts as a front end to multiple local DBMS or is structured as a global system layer on top of local DBMS. The global system provides full database functionality and interacts with local DBMS at their external user interface. The global system provides some means (global schema or multidatabase language) of resolving the differences in data representation and function between local DBMS" (p.5). A multidatabase provides the ability to solve the communication diversities, help

the data interchanges between local heterogeneous databases, and reinforce the data accessibility without ruining the local databases' autonomy.

According to Litwin, cited by Davydov (1997), "MDBS is a system for managing information exchange and enforcing cooperation among several databases without a global schema". This approach was initiated as an effort to develop an advanced architecture "supporting dynamic exchange of information objects in real time among multiple databases". It is also the approach this paper is going to use to integrate the heterogeneous databases.

3. MDBS Approach

Compared to other approaches, multidatabase approach is a so-called loosely coupled database integration framework (Davydov, 1997). The dependencies between local databases are minimized so that each local database can remain the maximum autonomy. This system is supposed to provide two functions for data management (Litwin, 1993, p.19):

- a multidatabase manipulation language for queries (and updates) to more than one database, and
- possibly, a language for the definition of interdatabase dependencies.

The overall language of an MDBS for both data definition and manipulation was called a multidatabase language. In general, a global user accesses data from local databases through global users and data from the local databases (Pissinou and Vanapipat, 1996). The MDBS must provide an environment so that if conflicts with other users occurred, it would affect only the global level. The user remains the ability to preserve his own needs in priority. Therefore, the MDBS is intended to overcome the inconsistencies between local DBMS and support the environment transparently.

There are many potential inconsistencies existed in integrating heterogeneous databases. *Data conflicts*, *Consistency*, and *Concurrent control*.

3.1 Data conflicts

Disparate data have widely varying names and definitions. Most names are abbreviated and short. Most definitions are short, truncated sentences. They seldom provide explicit, precise, and thorough understanding of the data. Hence, the first issue that needs to be solved is the data representation differences. "The data shared over the medium must be the official data variations. If the data source does not have the official data variation, it must translate its non-official data variation to an official data variation by accepted data translation schemes before sharing. If the data targets do not use the official data variation, they must translate the official data variation to their non-official data variation by accepted data translation schemes" (Brackett, 1994, p.20).

To solve the name difference problem, a set of unified naming rules is used in multidatabase level to uniquely identify all data stored in different local databases. This approach uses a unique name to identify synonyms and gives homonyms identifiable names. The global system uses its corresponding table to access data stored in local databases. Thus, the name differences will not exist in multidatabase level. The data names used in multidatabase must be consistent, unique, and meaningful although it may relate to several names in different local databases.

Data structure schemata will be needed for multidatabase to convert data between local databases. A data structure schema shows "the relationships between data subjects and the content of each data subject in the formal data resource" (Brackett, 1994, p.91). Heterogeneous databases may have variant data structures for the same data. While global users access local databases, MDBS converts the data accordingly so that the local DBMS can recognize and dispose of the data. Detail data translation process is introduced in Brackett's *Data Sharing: Using A Common Data Architecture* (1994).

Singh et al., (1997) proposed two functions that a MDBS must have to validate the data. The functions are referred to as *Data Purification*. (p.217)

- *Specification of complex validation rules and queries*. Data is validated based on the rules that either defines whether the given data is correct or is suspected to be incorrect. Processing each rule may require complex operations such as joining, selection, and aggregation.
- *Rapid refinement of validation specification*. The system verifies correctness and cleanliness of data with respect to specifications of what is valid. Typically, it takes several iterations to determine the right specification.

Data purification assures the data used is valid in format, value, and time.

3.2 Consistency

Another important problem of semantic interoperability in a multidatabase environment is maintaining the consistency of interdependent data physically stored in multiple databases (Pissinou and Vanapipat, 1996, p.36). Data updates may fail because of inconsistencies among copies; and multiple updates may cause integrity collisions (Olson, 1995). Breitbart, Silberschatz, and Thompson (1989) proposed a theorem in order to ensure the consistency of multidatabases. It says that the consistency of a global database is assured if the following conditions hold:

- Any local transaction from a set of local transactions is a read-only transaction or can write only non-replicated data items.
- There exists at least one selection of sites for executing global operations for transactions from a set of global transactions such that the corresponding transaction graph for the set of global transactions is acyclic.

Thorough descriptions and the proof can be found in Breitbart, Silberschatz, and Thompson (1989). The methodology is used in MDBS so that the consistency of the data can be ensured.

3.3 Concurrency control

A transaction can be divided into a series of reading operations and writing operations on a database. A centralized scheduler is used to perform transaction serialization. All unexecuted operations are put into a repository named *history*. The operations in this history are rearranged as a stream of interleaved operations. There are rules to perform this rearrangement. The scheduler orders the operations in a manner that conflicts are avoided and the integrity of the transactions are maintained. Thus, when DBMS executes the operations accordingly, the concurrency problems will not happen. Detail algorithms are depicted in Bobak (1996).

This approach provides a means to solve the concurrency problems, for both local DBMS and MDBS. However, the conflicts may occur when both DBMS and MDBS are manipulating the local database. The global system has enough information to provide concurrency control for global transactions, but it does not have information about local transactions. Therefore, it cannot provide total concurrency control. "Updates must be performed through the local DBMS interface on a node-by node basis" (Bright et al. 1993a).

The three aforementioned issues are the underlying impediments of MDBS. They are all related to the essences of data. In a nutshell: MDBS is not a single product you buy or a system you build, but rather a highly focused architectural framework for data integration (Davydov, 1997). Many other issues are not included for their essences are beyond the discussion of this paper. However, this study provided good directions and issues to databases integration.

4. Conclusion

"The multidatabase approach was intended as a new methodology for the design of database systems, in particular distributed databases. The general idea was that such systems should be able to manage collections of autonomous databases without a global schema and control. The methodology proposed new basic concepts, a general architecture and several new capabilities for relational languages" (Litwin, 1993, p.38). This paper has discussed several solutions in solving the problems for integrating heterogeneous databases. Davydov (1997) once defined the role of a MDBS as "a new kind of database environment used as a foundation for enabling sophisticated, multifunctional systems that can give organizations a significant edge over the competition" (p.63). It is also the intention of this paper.

However, there are still many other issues beyond the discussion of this paper. However, they are as important as the aforementioned issues. Without discreet controls, they are very much capable of causing the whole global system to fail functioning. Here are some issues.

Recovery control is to provide data protection and recovery procedures for all transactions (Gligor and Luckenbaugh, 1989). A MDBS without recovery control may result in disaster when data conflict or system function failure happens. Network bottleneck influences MDBS a lot in the way that frequent global accessing depends heavily on the message transportation time and accuracy. Security issue is also critical because a MDBS needs enormous authentication to access local databases. Failing to control the authentication may result in letting outsiders take advantage of this and to intrude the local databases. Another problem facing the existing multidatabase solutions is their weak expressiveness for the representation of the temporal changes, as well as the temporal constraints, and the representation of events in the external world (Pissinou and Vanapipat, 1996, p.36). In addition, the feasibility as well as the performance is not included in the discussion.

Singh et al. (1997) stated that heterogeneous database systems must address the following aspects:

- Getting to the data with a high-level view.
- Coordinating transactions across systems.
- Data cleaning
- Fusing traditional data with nonstandard data.

These four aspects directly relates to the essences of the data which are the main concern of integration of heterogeneous databases. Brackett (1994) stated that "Data sharing is not really about sharing data. It's about people understanding each other's views of data and the real world those data represent. It's about people using common data to meet different and changing business needs. It's about people working together to refine disparate data into a formal data resource and developing a mature data resource. It's about people combining available resources to meet high demands for information. People, not technology, make shared data a reality" (p.355). Current research in integrating databases using neural-networks can be found in the works of Wen et al. (2000).

This paper has provided the issues of a MDBS and the inconsistencies of integrating a heterogeneous database to a MDBS. When developing a fully functional MDBS, database architects and analyst should thoroughly look at these issues to avoid future pitfalls of the system. The solutions provided in this paper will significantly help in constructing and designing a MDBS while avoiding the problems that may arise shortly after development.

5. Suggestions for Future Research

In the constant improvements of technologies and systems, database integration remains a focus of study with the web and globalization. The internet has provided a means to access information globally, in real-time and at remote locations. This is possible when a MDBS is fully functional and capable of querying other MDBS. Future researchers have the option to investigate on the reliability of such functional systems, performance, feasibility and security issues of MDBS.

References

1. Anne Le Calve & Jacques Savoy (2000). Database Merging Strategy based on logistic regression. *Information Processing & Management*. 36 (3), pp. 341-359.
2. Appleton, D. S. (1989). The Technology Of Data Integration. *Integration Of Information Systems: Bridging Heterogeneous Databases* (pp. 5-10). New York: Institute Of Electrical And Electronics Engineers. (Originally printed in *Datamation*. Nov, 1985. pp. 106-116).
3. Batini, C., Lenzerini, M., & Navathe, S. B. (1989). A Comparative Analysis Of Methodologies For Database Schema Integration. *Integration Of Information Systems: Bridging Heterogeneous Databases* (pp. 72-112). New York: Institute Of Electrical And Electronics Engineers. (Originally printed in *ACM computing Surveys*. Vol. 18, No. 4, Dec. 1986).
4. Bobak, A. R. (1996). *Distributed and Multi-Database Systems*. Norwood, Massachusetts: Artech House.
5. Bouguettaya, A., & Milliner, S. (Nov. 1995). Co-database Approach to Database Interoperability. *IEICE Transactions On Information And Systems* (Vol. E78-D. No. 11, pp. 1388-1395). Tokyo, Japan: Institute of Electronics, Information and Communication Engineers.
6. Bourbakis, N. G., Marinos, L., & Papazoglou, M. P. (1993). Distributed Heterogeneous Information Systems & Schema Transformation. *Multidatabase Systems: An Advanced Solution For Global Information Sharing* (pp. 189-198). Los Alamitos, California: IEEE Computer Society Press. (Originally printed in *PARBASE-90: Proc. International Conference. On Databases, Parallel Architectures, And Their Applications*. 1990. pp. 388-397).
7. Brackett, M. H. (1994). *Data Sharing: Using A Common Data Architecture*. USA: John Wiley & Sons, Inc.
8. Breitbart, Y. (1993). Multidatabases Interoperability. *Multidatabase Systems: An Advanced Solution For Global Information Sharing* (pp. 41-48). Los Alamitos, California: IEEE Computer Society Press. (Originally printed in *SIGMOD Record*. Vol. 19, No. 3, Sep. 1990. pp. 53-60).
9. Breitbart, Y., Silberschatz, A., & Thompson, G. (1989). Transaction Management In A Multidatabase Environment. *Integration Of Information Systems: Bridging Heterogeneous Databases* (pp. 135-143). New York: Institute Of Electrical And Electronics Engineers.
10. Bright, M. W., Hurson, A. R., & Pakzad, S. H. (1993a). A Taxonomy and Current Issues in Multidatabase Systems. *Multidatabase Systems: An Advanced Solution For Global Information Sharing* (pp. 3-12). Los Alamitos, California: IEEE Computer Society Press. (Originally printed in *Computer*. Vol. 25, No. 3, Mar. 1992. pp. 50-60)
11. Bright, M. W., Hurson, A. R., & Pakzad, S. (1993b). *Multidatabase Systems: An Advanced Solution For Global Information Sharing*. Los Alamitos, California: IEEE Computer Society Press.
12. Cole, B. (Jun. 5, 1995). New Middleware Pushing Beyond Data Access. *Network World* (Vol. 12, No. 23, pp. 1 & 64). Framingham, Massachusetts: CW Communications.
13. Darling, C. B. (Nov. 1995). Dig Deep To Strike Data Access Gold. *Datamation* (Vol. 41, No. 20, pp. 76-81).
14. Davydov, M. M. (Aug. 1997). Bridging The Data Archipelago. *Database Programming And Design* (Vol. 10, No. 8, pp. 54-63). San Francisco. California: Miller Freeman Publications.
15. Fang, D., Hammer, J., & McLeod, D. (1993). The Identification And Resolution Of Semantic Heterogeneity In Multidatabase Systems. *Multidatabase Systems: An Advanced Solution For Global Information Sharing* (pp. 52-59). Los Alamitos, California: IEEE Computer Society Press. (Originally printed in *First Int'l Workshop On Interoperability In Multidatabase System*. 1991. pp. 136-143).
16. Gligor, V. D. & Luckenbaugh, G. L. (1989). Interconnecting Heterogeneous Database Management Systems. *Integration Of Information Systems: Bridging Heterogeneous Databases* (pp. 35-45). New York: Institute Of Electrical And Electronics Engineers. (Originally printed in *Computer*. Jan. 1984. pp. 33-43).
17. Grosky, W. I., and Sirounian, L. (Feb. 1995). A Knowledge Model For Unifying Deductive And Non-Deductive Heterogeneous Databases. *IEEE Transactions On Knowledge And Data Engineering* (Vol. 7, No. 1, pp. 82-105). New York: Institute Of Electrical And Electronics Engineers.
18. Hutchinson, N., Notkin, D., Sanislo, J., & Schwartz, M. (1989). Heterogeneous Computing Environments: Report On The Acm Sigops Workshop On Accommodating Heterogeneity. *Integration Of Information Systems: Bridging Heterogeneous Databases* (pp. 24-32). New York: Institute Of Electrical And Electronics Engineers. (Originally printed in *Communications Of The ACM*. Vol. 30, No. 2, Feb. 1987).

19. Joon, H.L. (1998). Combining the evidence of different relevance feedback methods for information retrieval. *Information Processing & Management*. 34 (6), pp. 681-691.
20. Klaus-Dieter, Schewe & Bettina, Schewe (2000). Integrating Database and Dialogue Design. *Knowledge and Information Systems*. 2 (1), pp. 1-32.
21. Litwin, W. (1993). From Database Systems to Multidatabase Systems: Why and How. *Multidatabase Systems: An Advanced Solution For Global Information Sharing* (pp. 13-40). Los Alamitos, California: IEEE Computer Society Press. (Originally printed in *Proc. Sixth British National Conference on Databases*, pp.161-188. New York: Cambridge University Press. 1988).
22. Olson, J. (Jun. 1995). Building A Database Topology Strategy. *Database Programming And Design* (Vol. 8, No. 6, pp. 52-61). San Francisco. California: Miller Freeman Publications.
23. Palmer, J., Ray, C., & Wohl, A. (1995). *Office Automation: A System Approach* (3rd edition). Cincinnati, Ohio: South-Western Educational Publishing.
24. Pissinou, N. and Vanapipat, K. (Jan. 1996). Active Database Rules In Distributed Database Systems: A Dynamic Approach To Solving Structural And Semantic Conflicts In Distributed Database Systems. *Computer Systems: Science & Engineering* (Vol. 11, No. 1, pp. 35-44). London: Butterworth Scientific Ltd.
25. Seung-Jin Lim & Yin-Kai, Ng (2001). A Hybrid Fragmentation Approach for Distributed Deductive Database System. *Knowledge and Information Systems*. 3 (2), pp. 198-224.
26. Singh, M. P., Cannata, P., Huhns, M. N., Jacobs, N., Ksiezyk, T., Ong, K., Sheth, A. P., Tomlinson, C., & Woelk, D. (Apr. 1997). The Carnot Heterogeneous Database Project: Implemented Applications. *Distributed And Parallel Databases* (Vol. 5, No. 2, pp. 207-225). Boston. Massachusetts: Kluwer Academic Publishers.
27. Sippl, R. (Jan-Feb. 1996). SQL Access Group's call-level interface. *Dr. Dobb's Journal* (Vol. 21, No. 13, pp. 11-14).
28. Stonebraker, M. (1993). Future Trends In Database Systems. *Multidatabase Systems: An Advanced Solution For Global Information Sharing* (pp.339-350). Los Alamitos, California: IEEE Computer Society Press. (Originally printed in *IEEE Transactions On Knowledge And Data Engineering*. Vol. 1, No. 1, Mar. 1989. pp. 33-44).
29. Wen-Syan Li, Chris, Cliffon, & Shu-Yao-Liu (2000). Database Integration Using Neural Networks: Implementation and Experiences. *Knowledge and Information Systems*. 2 (1), pp. 73-96.
30. Leslie Roni Leong, Yao-Chin Huang © 2001.

Notes