Towards the Use of both Financial and Non-financial Data for Decision Making: A Conceptual Framework for Federated Database Systems

David X. Zhu, PhD
Assistant Professor
Department of Accounting and Finance
College of Business Administration
California State University, Stanislaus
Turlock, CA 95382, USA.

Sijing Zong, PhD
Associate Professor
College of Business Administration
California State University, Stanislaus
Turlock, CA 95382, USA.

Abstract

Most of the data sources in business and academia are financial data. Non-financial data are difficult to collect, integrate, and retrieve and are in most cases stored as separate and heterogeneous data sources that could not be readily used in financial analysis and research. However the recent development in finance research has put significant attention on the behavioral aspects of the participants of business and financial activities, which in turn requires the inclusion of the non-financial data that are related to such activities. One of the possible solutions is the federated database, which is a collection of distributed databases that are heterogeneous and autonomous in nature. This research extends previous works on federated database systems and proposes a conceptual framework for an FDBS (federated database system) that facilitates centralized control, as in a centralized database system, over distributed heterogeneous databases with the capabilities of autonomously discovering and retrieving data from dispersed internal and external database systems. A comparison of centralized, distributed, and federated database systems is also provided.

Introduction

Accounting/financial data have been widely used in academic research where COMPUSTAT, CRISP, and Datastream are examples of well-established accounting/financial databases.Meanwhile, non-accounting/financial data played an important role in research, but lack integrated databases. Most of the time, researchers have to collect needed non-accounting/financial data by hand. The hand collection process could be very time consuming and be difficult to avoid errors even when extra cautious were used. This lack of data or the high cost of non-accounting/financial data may have hindered research in various subjects. Schiller (2003) reviewed the evolution of behavioral finance research that argues financial markets are inefficient. Traditional finance theories believe that financial markets are efficient. It was not until the late 1990s that behavioral finance started to be acknowledged by the mainstream. He points out that the non-financial data that are ignored by traditional finance theories may have played significant roles in the inefficiency of the market.

Data is not only needed in academic research but also in the decision making of all sorts of businesses. Firms need data, both financial and non-financial, for strategic and operational decisions. Due to the high level of competition and fast changing business environment, firms cannot afford to wait for a lengthy data collection to obtain needed data for their decisions. It is critical that firms receive needed data, both financial and non-financial, in a timely manner. Other than utilizing previously mentioned popular academic accounting/financial data bases, many large enterprises maintain their data in various distinct independent databases that have been developed at different times on different platforms and database management systems (DBMSs). Additionally, due to increasing globalization, mergers, and acquisitions it is inevitable that different portions of many of these enterprises will use various heterogeneous systems to store and retrieve their critical data. For these reasons, along with the evolution of information technologies such as the Internet, the diversity of data and information formats continues to expand. Without the capability of combining all the available data from these systems it is understandably difficult for the enterprises to make fully informed decisions.

This underutilization of available data sources is a contributing factor to sub-optimization in the business decision-making environment. Due to prohibitive financial and organizational costs, enterprises cannot afford to replace the existing disparate databases with a common system. Hence, the need arises to integrate the functions of the enterprise and their information systems along with the databases they are based on. Users and application programs should be able to access data as if they were stored in a homogeneous and consistent database (Thiran, et al., 2000). Integrated with other features, functionalities, and capabilities to retrieve and process external data, this type of system has been termed a federated database system (FDBS). This concept has also occasionally been referred to as virtual database technology (Prasad and Rajaraman, 1998; Rajaraman, 1998; Gupta et al., 1997) and enables the Web and other external data sources to behave as an extension of an enterprise's internal database management systems (Prasad and Rajaraman, 1998). This federated database system is often enabled by a centralized system that controls and manages relationships and interactions among distributed information sources in order to process queries (Panti, et al., 2002).

FDBS technology has drawn interest from industry. One of the first commercial products associated with the FDBS concept is DataJoiner, middleware that IBM introduced in 1995. Since then software venders have been providing applications that implement this technology. For example, the Microsoft SQL Server 2000 has a "Federated Database Servers" feature that enables a group of SQL Servers to share processing load and allow the users to scale a set of servers to support exceptional processing needs. Exponential advances in computer technologies are producing data of unprecedented quantity and quality. Multi-terabyte and even petabyte (1000TB) data collections are emerging as major assets to modern enterprises. Additionally, dramatic improvements in storage and network performance also drive the demand for systems that can integrate, correlate, and mine distributed data (Foster and Grossman, 2003). The traditional centralized DBMSs and distributed DBMSs do not have sufficient features to accommodate and fully utilize these phenomenal advances. Further, collecting and processing heterogeneous data from different sources using the existing DBMSs and applications requires a huge amount of time and labor making obtaining integrated information very difficult and time consuming. Incorporating artificial intelligence (AI) capabilities into the system to autonomously search and process the data may aid in solving these problems.

This paper introduces a framework for integrating the concepts of centralized DBMS and distributed DBMS taking advantage of both in terms of centralized control and access to distributed heterogeneous data sources. The AI components are added to the framework to enable autonomous data discovery in a heterogeneous environment. In this framework, centralized control does not mean that the federated architecture has control over the data on the component database systems (DBSs); instead, it provides a mechanism of controlling the data retrieved and transferred from the component DBSs. This paper is organized in the following manner: first, basic FDBS concepts are introduced. A framework is then proposed illustrating how a federated database system can provide centralized control and autonomous searching in a heterogeneous and distributed data source environment. Further discussion of this framework is undertaken and some future research directions are also addressed.

BACKGROUND

Early studies of the concept of FDBS began in the late 1970's, when this term was coined in the literature (Sheth and Larson, 1990). Most present research has focused on highly technical aspects and on algorithm development. FDBS has its initial roots in investigations into distributed database systems. Several different definitions of FDBS exist. Some suggest that a FDBS is simply a distributed database, the components of which are physically stored in a number of distinct real databases at a number of distinct sites (Sadanandan and Chakravarthy, 1994). Others simply define FDBS as an application that interacts with other enterprise systems via database APIs, EAI/EDI messages, XML messages, distributed objects or through its own "proxy" agents (Blackham, et al., 2001). Sheth and Larson (1990) define it as a collection of cooperating database systems that are autonomous and possibly heterogeneous.

Significant work is done by Sheth and Larson (1990), who define a reference architecture for distributed DBMSs from system and schema viewpoints and show how various FDBS architectures can be developed. Further study was conducted by Gardarin et al. (1995), who introduce mechanisms for federating object-oriented and relational databases. The authors suggest that the system adopts the federated approach to database interoperability to overcome the limitations of traditional homogeneous distributed DBMSs. This adds layers of software to pre-existing heterogeneous DBMSs without privileging one system.

Additionally, this system does not violate the autonomy of the pre-existing databases; that is, databases participating in a federation do not lose control over their data and administrators ideally do not have to restructure their systems in order to participate in the federation. Pitoura et al. (1995) studied a federation of pre-existing distributed, heterogeneous, and autonomous database systems they termed a multidatabase system. They provide a concrete analysis and categorization of the various ways in which object-orientation has affected the task of designing and implementing these types of systems. A FDBS may be characterized along three dimensions: distribution, heterogeneity, and autonomy (Sheth and Larson, 1990). Distribution implies that multiple databases may be stored on a single computer system or on multiple computer systems interconnected by communication systems. Data may be distributed among these multiple databases in different ways including vertical and horizontal database partitions. Heterogeneities may results from structural or technical issues as well as semantic differences. Autonomy refers to the degree that database systems are under separate and independent control and data is shared only as long as the autonomous database systems retain control of their own data. Three component autonomy types are identified as design, association, and execution autonomy.

Egyhazy (2000) notes that the "investment in existing databases is enormous, and a critical factor in their potential reuse in tomorrow's distributed and interoperable environments lies in the development of software architectures, and tools for the resolution of the database heterogeneity problem." Three primary sources of database heterogeneity are identified as data models, data values, and logical representations. He further states that the heterogeneous database problem arises primarily when individual queries require access to multiple autonomous databases. The practical importance of integrating heterogeneous databases consists of enabling coexistence of different DBMSs that support various data models meeting specific application requirements. Kalinichenko (1990) argues that this integration requires the introduction of a DBMS-independent generalized level of data representation and manipulation along with a generalized model of a corresponding integration system. He proposes an approach for transformation of heterogeneous data models utilizing equivalent data model mapping construction. Additionally, Wache and Stuckenschmidt (2001) introduce context transformation theory to tackle the heterogeneity problems and address the issues in practical context transformation for information system interoperability.

Evolution of Models

Sheth and Larson (1990) provided the foundation for developing models for FDBSs. They propose a FDBS reference architecture consisting of data, databases, commands, processors, and schemas and mappings. Processors and schemas "play especially important roles in defining various architectures." Processors are defined as software modules that manipulate commands and data; while schemas are descriptions of data managed by one or more DBMSs.

In the reference architecture, Sheth and Larson (1990) identify four types of processors. As shown in Figure 1, these processors each perform different functions on data manipulation commands and accessed data and may work independently. These processors may also be employed in a cooperative manner as shown in Figure 2, adding logical interrelationships among these processors. The five-level schema architecture of a federated database system includes the local schema (conceptual schema of a component DBS), the component schema (which is derived by translating local schemas into a data model called the canonical or common data model), the export schema (which represents a subset of component schema that is available), the federated schema (an integration of multiple export schemas), and the external schema (which defines a schema for a user and/or an application or a class of users/applications). The simplified architecture using these schemas is illustrated in Figure 3.

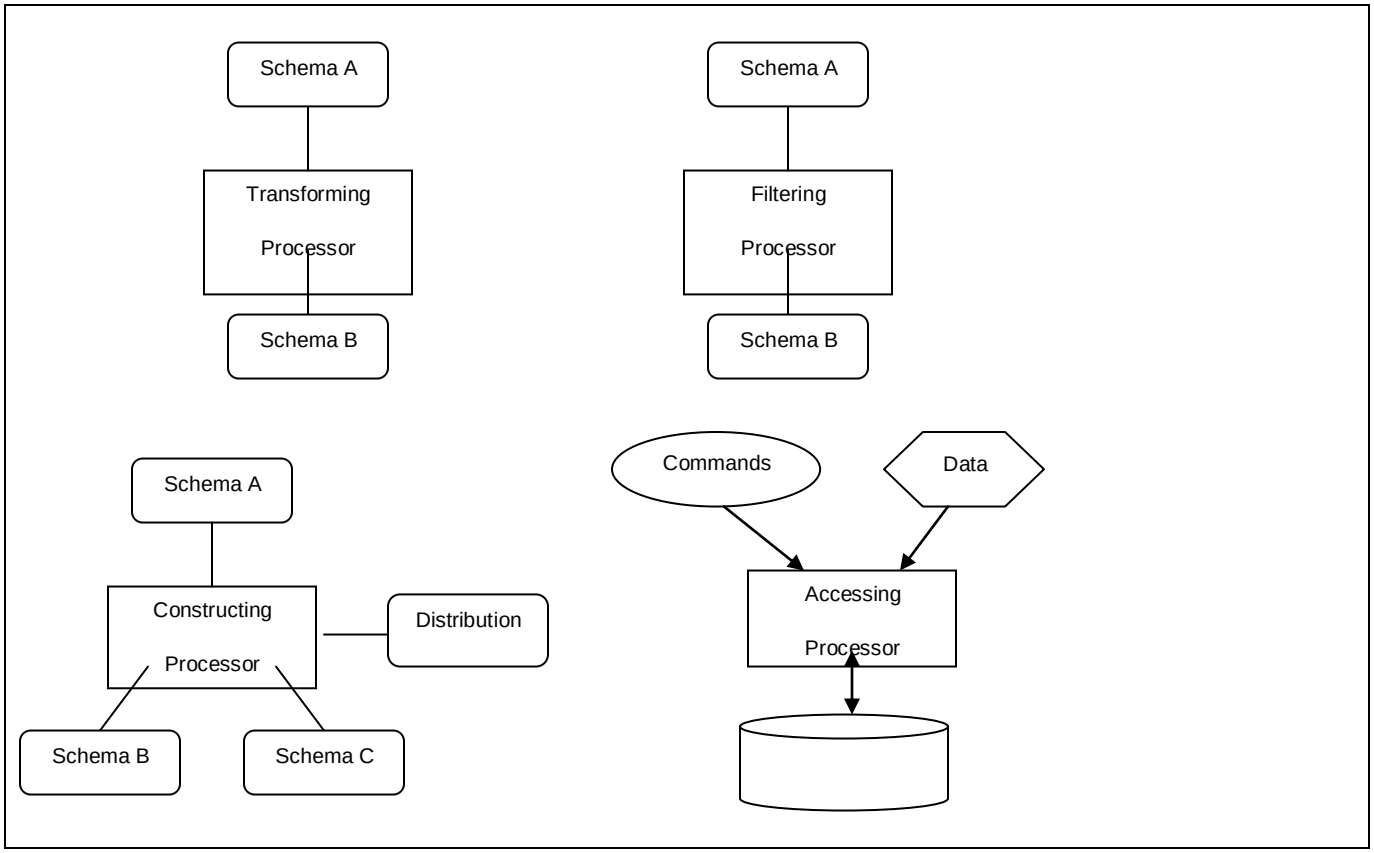

Figure 1: Processor types in reference architecture of Sheth and Larson (1990)

Nijenuis et al. (1997) report issues related to the use of workflow management systems and FDBSs to support business processes that operate on large, heterogeneous, and autonomous collections of information systems. The integrated architecture suggested is based on the OSCA architecture (Bellcore 1992), which describes a three layer architecture: the data layer (which groups the corporate data management functionality), the processing layer (which contains business operations and management functionality), and the user layer (which contains the human interaction functionality). A FDBS is positioned in the data layer as a uniform access mechanism for accessing the data layer components, shown in Figure 4.

Other approaches have also been proposed. Gardarin et al. (1997) worked on the IRO-DB project and developed tools for integrated access of relational and object-oriented databases and for designing and maintaining integrated applications on large federations of heterogeneous databases. One of the major contributions of the authors is to identify seven problems associated with designing a FDBS: choosing a pivot database model, organizing schemas and repositories, identifying objects, choosing an interchange protocol, managing integrated objects, optimizing distributed queries, and managing distributed transactions.

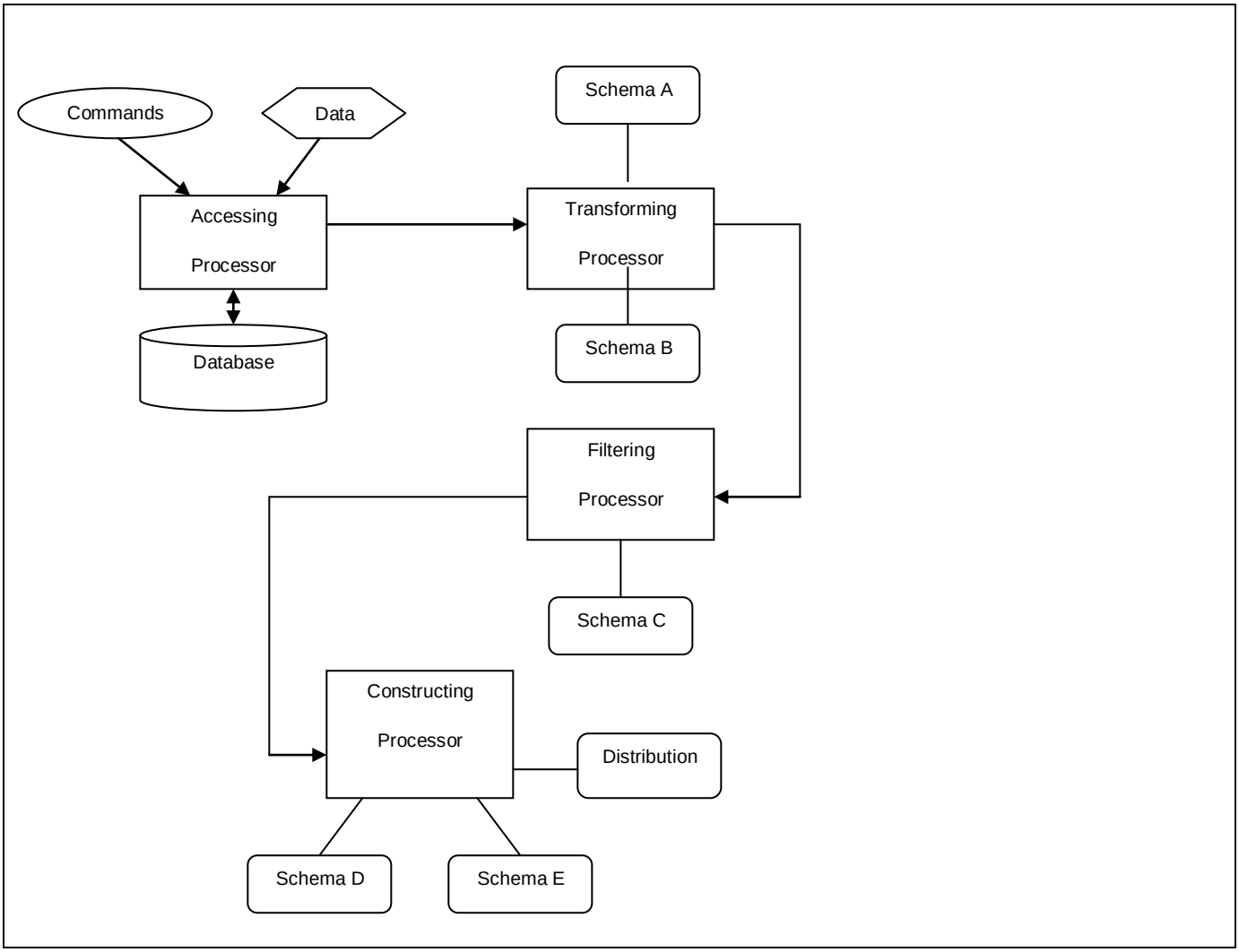

Figure 2: Simplified diagram of logical relationships among processors

Gardarin et al (1997) define the FDBS architecture with three layers, but with different names, which are: local layer, communication layer, and interoperable layer. This architecture is shown in Figure 5.

The InterDB architecture was developed by Thiran et al. (1998). This architecture is based on four defined schemas:

- Local Physical Schema (LPS) (extracted according to a common abstract physical model)
- Local Logical Schema (LLS) (obtained by cleaning and enriching the physical schema)
- Local Conceptual Schema (LCS) (interprets the logical structure by extracting their underlying semantics)
- Global Conceptual Schema (GCS) (includes the semantics of the local schemas)

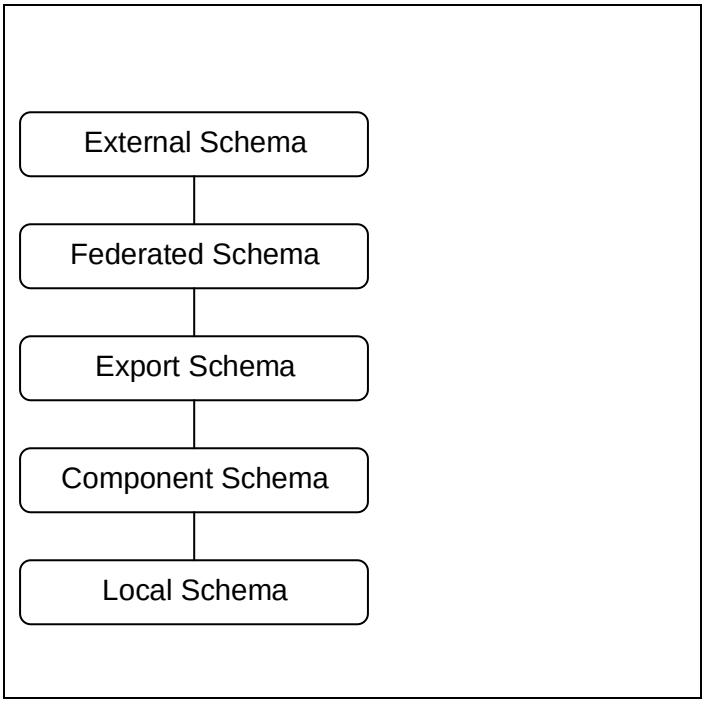

Figure 3: Simplified five-level schema architecture

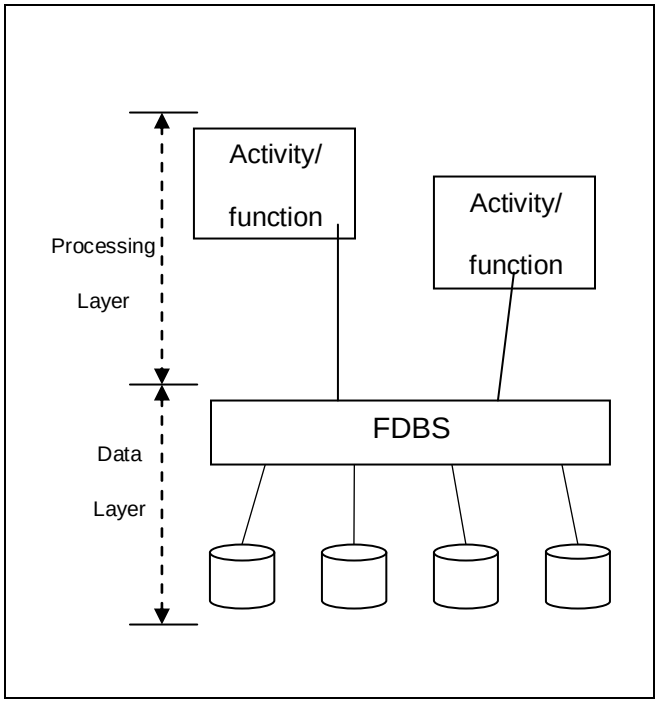

Figure 4: OSCA-based FDBS architecture

A FDBS' ability to integrate data across geographically distant resources would be expected to be greatly enhanced by dramatic improvements in network performance. Distributed system middleware allows distributed communities, or virtual organizations, to access and share data, networks, and other resources in a controlled and secure manner. Advances in these areas promise to provide required capabilities. Foster and Grossman (2003) provide three different scenarios to illustrate applications impossible during the first few years of the new millennium but achievable over advanced networks that are quickly becoming more available. In the *virtual data warehouses* scenario, high-speed optical networks make it possible for data to be stored at its sources rather than centralized. Reports can be generated upon demand by merging from multiple sources with most recent data as long as bandwidth is available. In the business world, with all-optical networks and distributed data services, it becomes feasible to consider replicating transformations from core systems to remote backup systems for business continuity. The third scenario addresses large data sets. Instead of performing data processing offline and preparing and distributing data periodically, with optical networks and data-integration services, large data sets can be continuously updated and users always have access to current data.

Due to its systematic heterogeneous nature, the FDBS needs to address the problem of interoperability of database systems or information systems in general. Wrappers have been described as a method of achieving this. A wrapper (or translator) is a component that converts queries and data from one format to another. Wrappers are typically utilized to provide integrated access to heterogeneous information sources (Papakonstantinou et al. 1995). When an application issues queries in a standard query language the wrapper for each source converts the query so that it is understandable by the underlying data sources. The wrapper receives the results and translates them into a format understood by the application. A wrapper can perform three types of processes: directly supported queries, logically supported queries, and indirectly supported queries (Papakonstantinou et al. 1995).

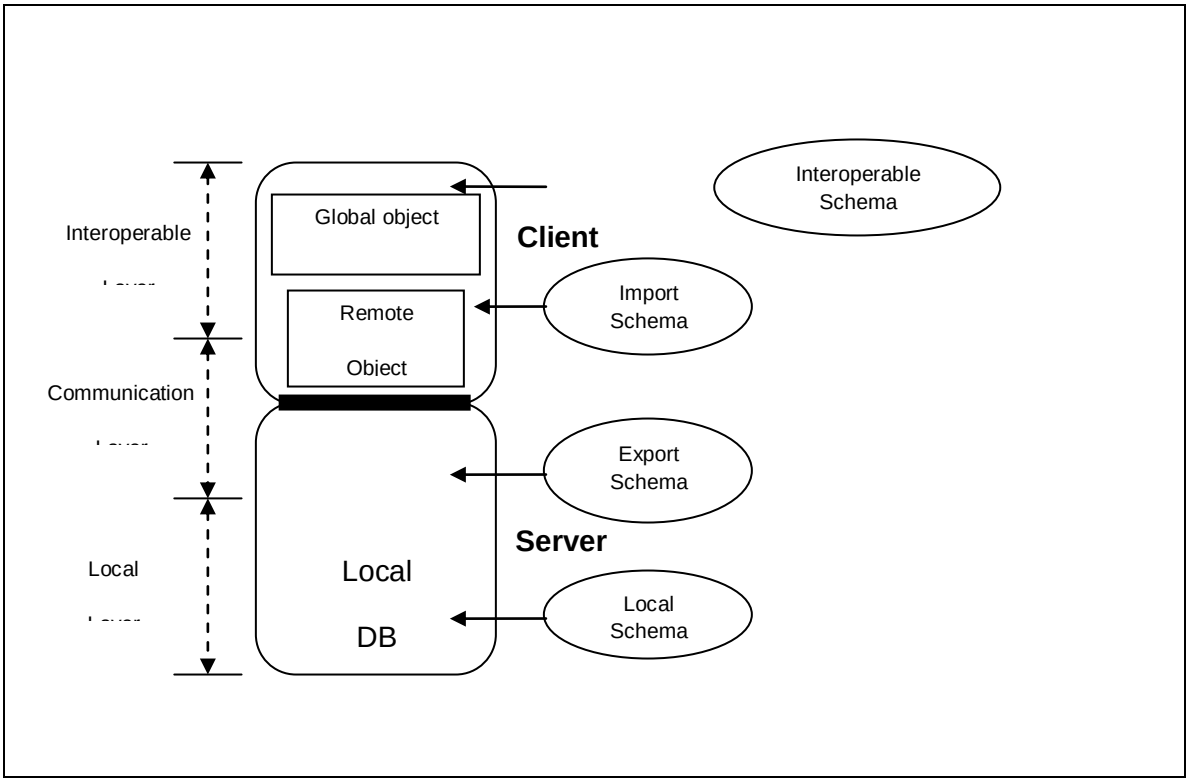

Figure 5: Simplified IRO-DB schema architecture

THE FRAMEWORK

Based on the evolution of the system architecture for FDBS, the framework proposed for federated database systems (Figure 6) is built on four main schemas and six layers. The schemas are consistent with those of Sheth and Larson (1990) reference architecture with a minor alteration of encapsulating the component schema within the export schema. The development of the layer structure is a logical and conceptual integration of the works by Gardarin (1997), Nijenhuis et al. (1997), Thiran et al. (1998), Thiran et al. (2000), and Foster and Grossman (2003). This was informed additionally by the advances in the theory and application of agent-based technologies. The following sections will look into each segment of the framework. These segments are based on the schema hierarchy (data, export, federated, and external schemas) and incorporate the layer sub-structure.

Data Schema

The data schema, with the network components added to it, consists of the internal and external data layers. Each component data source has a conceptual local schema which is expressed in the native data model of the component DBMS and hence different local schemas may be expressed in different data models. The internal local DBSs are tightly coupled and are owned by the enterprise that implements the FDBS framework, which may be centralized or distributed logically or geographically. This layer has a relatively stable structure over time and the data managed by this layer generally has an enterprise-wide scope. Changes in the architecture of the internal data layer are typically related to business changes; examples include new products, services, customer types, etc. The access to these internal local DBSs is through the enterprise local area network (LAN) providing a low latency connection with limited bandwidth (often Ethernet with 100mbps). In this layer, the enterprise has full and/or partial control over these DBSs.

Unlike the internal DBSs, the external DBSs are those outside of enterprise boundary. These may include such heterogeneous data sources as the Internet and sources from associated enterprises or organizations such as vendors and industry groups, virtual enterprises, or others. The enterprise has limited or no direct control over these DBSs and thus they are loosely coupled. To facilitate this layer an optical backbone may be required for fast connection and large data transformations. The data sources from the scenarios described in Foster and Grossman (2003) can also be categorized into this layer.

Export Schema

The export schema in this framework is the combination of component and export schemas as described in Sheth and Larson (1990). The three major functionalities of this schema are data source discovery, transformation, and uploading to the FDBS upper layer. These functionalities are implemented by wrappers and agents.

Wrapper / Agent Dispatcher

A wrapper is a software component that provides access to heterogeneous information sources by converting application queries into source specific queries or commands. The wrappers describe the divergent local schemas using a single representation and can add the semantics that are missing in a data schema to the export schema. They can facilitate negotiation and integration tasks performed when developing a tightly coupled FDBS. Similarly, they may facilitate negotiation and specification of views and multi-database queries in a loosely coupled FDBS (Sheth and Larson , 1990). Essentially, wrappers are used to process schema translation from data schema to an export schema by generating the mappings between two schemas, in another words, they map incoming queries into native commands of the underlying source. The wrapper implementation toolkit developed by Papakonstantinou et al. (1995) may be used for rapidly building wrappers.

The wrappers may be stand alone software as described above for data transformation. However, they may also be autonomous and be implemented by using agent-based technology providing autonomous data discovery functionality for the FDBS.

Jennings and Wooldridge (1998) provide the definition of agent as is used in this discussion, "An agent is a computer system situated in some environment, and that is capable of autonomous action in this environment in order to meet its design objectives." Agents also have the following properties (Wooldridge and Jennings, 1995, p.2):

- *Autonomy*: agents operate without the direct intervention of humans or others, and have some kind of control over their actions and internal state;
- *Social Ability*: agents interact with other agents (and possibly humans) via some kind of agent-communication language (ACL);
- *Reactivity*: agents perceive their environment (which may be the physical world, a user via a graphical user interface, a collection of other agents, the Internet, or perhaps all of these combined) and respond in a timely fashion to changes that occur in it;
- *Pro-activeness*: agents do not simply act in response to their environment; they are able to exhibit goal-directed behavior by taking the initiative.

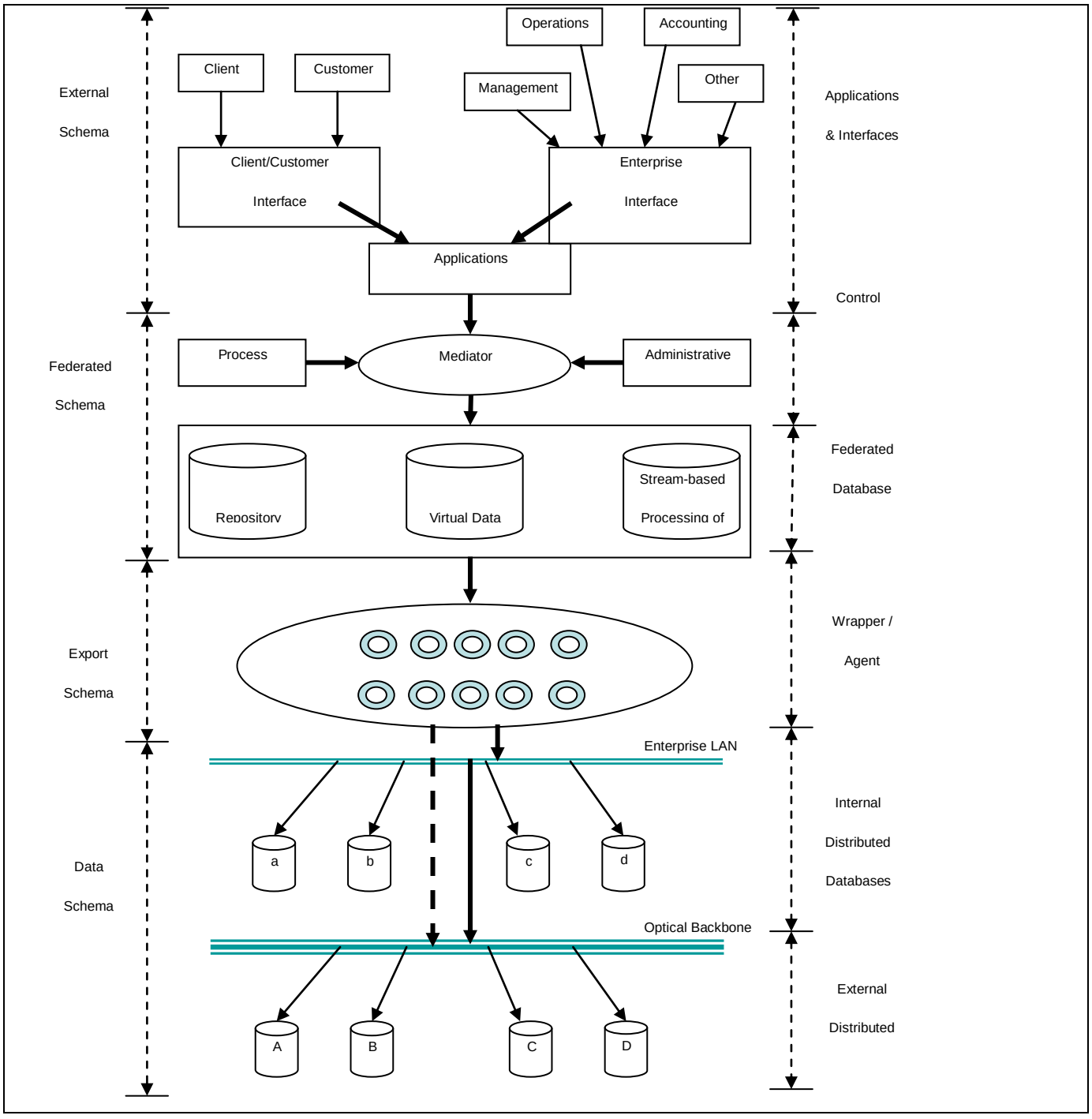

Figure 6: Federated Database System

This notion of an agent as being a self-contained and concurrently executing software process that encapsulates some state and is able to communicate with other agents via message passing is seen as a natural development of the object-oriented concurrent programming paradigm (Agha, 1986; Agha and Yonezawa, 1993). According to Lau and Wong (2001), there are five cooperative types of agent:

- *Coordinating agent*: inputs the requests of the partners and sends them out to other appropriate partners,
- *Communication agent*: is responsible for receiving and sending all kinds of message packets and ensures quick and safe communications among the agents dispatched and member databases (both internal and external),

- *Information agent*: deals with the decision-making using the knowledge and information from other agents,
- *Search agent*: assists in the search and collection of information from the underlying DBSs in the lower layers,
- *Reporting agent*: collects and sends information about the operational situation within the database system.

The agent-based wrappers can provide autonomous data discovery and transformation services to the export schema and pass the controls to the upper layers in the federated schema.

Federated Schema

A federated schema encompasses two layers: the federated database layer and the control layer. It is an integration of multiple export schemas and also includes the information on data distribution that is generated when integrating export schemas. Some systems use a separate schema called distribution schema or an allocation schema to contain this information (Sheth and Larson, 1990). The federated database layer is a collection of relational or objected-oriented databases including a virtual data warehouse that is constructed on the fly; stream-based processing of data that access large data sets that are continuously updated; and a repository that keeps copies of data exported from the lower layer that provides critical infrastructure for disaster recovery and business continuity. These databases are accessible and controlled by the mediator that resides in the applications and control layer.The mediator is a middleware software module that exploits encoded knowledge about some sets or subsets of data to create information for a higher layer of applications (Wiederhold, 1995). A mediator provides transparency to users by hiding details about the location and representation of relevant data to applications. Two types of controls are defined in this framework. Administrative control handles the maintenance and updates of the mediator. Process control controls the federated database through the mediator. This function includes data manipulation, access control, maintenance of the database and integrity constraints, updates, debugging, and administration, as well as security and policy management.

External Schema

The external schema defines a schema for a user and/or application or a class of users/applications. The layer associated with this schema is the applications and interfaces layer. The reasons for the use of external schemas are as follows (Sheth and Larson, 1990):

- *Customization*: a federated schema can be quite large, complex, and difficult to change. An external schema can be used to specify a subset of information in a federated schema that is relevant to the users of the external schema. They can be changed more readily to meet changing users' needs. The data model for an external schema may be different than that of the federated schema.
- *Additional integrity constraints* can also be specified in the external schema.
- *Additional access control:* Export schemas provide additional access control with respect to the data managed in the lower layer FDBS.

The applications include data mining tools that generate reports for different users such as clients and customers, as well as enterprise users on different levels and from different departments or subsidiaries. The major task of the applications is to provide data exploration and analysis. Data rendered accessible from the mediator can be analyzed in detail. These services can provide basic statistical summaries, enable visual exploration of data, and support standard exploratory functions, such as building clusters, computing the regression of one variable on another. Interfaces provide users convenient graphical tools for interaction with the FDBS. Access control is enforced on the applications as well as in the control layer. On the business side, the external users – clients and customers – can easily interact with the enterprise database through Internet or special network access arrangements such as EDI (Electronic Data Interchange). On the enterprise side, internal users can interact with the database through their PCs via networks such as a LAN or VPN (Virtual Private Network).

Both external and internal users are provided appropriate interfaces and access privileges. For example, top management level users may access the critical financial and personnel data with a DSS (Decision Support System) interface for decision making; operations managers may access the production control data for quality assurance and volume control; and customers may access the enterprise store-front on the Internet to obtain product information and support through web page interfaces with very limited access to the FDBS.

CONCLUSIONS AND FUTURE RESEARCH

This research has developed a conceptual FDBS framework, a logical architecture for realizing federations of centralized, distributed, and other federated heterogeneous database systems to provide centralized control over as well as autonomous data source discovery and intelligent data retrieval from dispersed database systems. The advances in network bandwidth capabilities enables the FDBS in the framework to access much larger data sources, utilize more unstructured data, and stream the data to the centrally controlled data warehouses and repository for real-time processing. In addition to accessing the enterprise-wide database systems, FDBS is able to discover and access external rich data sources such as millions of websites. Relevant technologies include web services mechanisms such as the Web Services Description language specifications, Grid-enabled data access and integration services, directory services (such as Lightweight Directory Access Protocol), XML and relational databases, semantic web technologies, and text-based web search mechanisms applied to unstructured text-based meta-data (Foster and Grossman, 2003).

Intelligent agents give life to the framework. The cooperation between the agents makes the autonomous data source discovery and data retrieval possible. Less human intervention in these processes makes the FDBS easy to maintain and operate. However, artificial intelligence is still at an infantile stage and the capability of agents still needs further development in terms of autonomy, social ability, reactivity, and pro-activeness. This paper has several limitations. Because the purpose of this research is to develop a conceptual framework, the technical details of each layer are not developed. To make the framework easier to implement in practice, further research needs to address the following issues: the interoperability of the heterogeneous DBMS at the data schema level, access control of the agents and wrappers to the lower level DBMS, technology-based limitations on agents in autonomously accomplishing their tasks, and the mechanisms managing the federated database. This final point includes the specification of integrity constraints, interoperability of different DBSs, data representation, and data mining techniques.

Other research issues may include: "How will the FDBS impact the business strategy and decision-making when more data is available with better accuracy?" "What financial implications will be, given the implementation of this framework may be complicated and costly?" "What other elements can be added to this framework?" "When implemented, will people accept it?" The technology acceptance model (Davis, 1989) may be used to test this. By answering these questions including others, the author believes that the study of FDBS will be further advanced.

REFERENCES

Agha, G. (1986). "ACTORS: A Model of Concurrent Computation in Distributed Systems", The MIT Press: Cambridge, MA

Agha, G., Wegner, P., and Yonezawa, A. (1993), "Research Directions in Concurrent Object-Oriented Programming", The MIT Press: Cambridge, MA

Bellcore (1992), "The Bellcore OSCA Architecture", *Technical Reference TR-STS-000915*, Issue 1, October 1992

Blackham, John; Grundeman, Peter; Grundy, John; Hosking, John; and Mugridge, Rick (2001), "Supporting Pervasive Business via Virtual Database Aggregation", *Proceedings of Evolve '2001, Pervasive Business, Sydney, Australia, DSTC*

Davis, Fred D. (1989), "Perceive Usefulness, Perceived Ease of Use, and End User Acceptance of Information Technology", *MIS Quarterly*, Vol. 13, pp. 318-339

Egyhazy, C.J. (2000), "From Software Resue to Database Resue", *International Journal of Software Engineering and Knowledge Engineering*, Vol. 10, No. 2, pp. 227-249

Embley, David W.; Campbell, Douglas M.; Liddle, Stephen W.; Smith, Randy D. (1998), Ontology-based Extraction and Structuring of Information from Data-Rich Unstructured Documents", in *the Proceedings of CIKM'98*

Foster, Ian, and Grossman, Robert L. (2003), "Data Integration in a Bandwidth-Rich World", *Communications of the ACM*, Vol. 46, No. 11, pp. 51-57

Gardarin, Georges; Finance, Beatrice; and Fankhauser, Peter(1997), "Federating Object-Oriented and Relational Databases: The {IRO}-{DB} Experience", *Conference on Cooperative Information Systems*, pp. 2-13

Gardarin, Georges; Gannouni, Sofiane; Finance, Béatrice; Fankhauser, Peter; klas, Wolfgang; Pastre, Dominique; Legoff, Régis; and Ramfos, Antonis (1995) "IRO-DB : A Distributed System Federating Object and Relational Databases", In *Object-Oriented Multibase Systems, Bukres, O., and Elmagarmid, A. Ed., Prentice Hall, Englewood Cliffs, N.J.*

Genesereth, M. R. and Ketchpel, S. P. (1994), "Software Agents", *Communications of the ACM*, Vol. 37, Issue 7, p48

Grunerta, Jens; Norden, Lars; and Weber, Martin (2005), "The Role of Non-financial Factors in Internal Credit Ratings", *Journal of Banking & Finance*, Vol. 29, No. 2, pp. 509-531

Gupta, Ashish; Harinarayan, Venky; Rajaraman, Anand (1997), "Virtual Database Technology", ACM SIGMOD Record, Vol. 26, No. 4, pp. 57-61

Haas, Laura, and Lin, Eileen (2003), "IBM Federated Database Technology", *IBM developerWorks on IBM website*

Jennings, N.R. and Wooldridge, M.J. (1998), "Applications of Intelligent Agents", *Agent Technology: Foundations, Applications, and Markets, Springer*, p3.

Kalinichenko, Leonid (1990), "Methods and Tools for Equivalent Data Model mapping construction", *in Proceedings of the International Conference on Extending Database Technology (EDBT '1990), Venice*

Katz, boris;, Felshin, Sue; Yuret, Deniz; Ibrahim, Ali; Lin, Jimmy; Marton, Gregory; McFarland Alton M.; and Temelkuran, Baris (2002), "Omnibase: Uniform Access to Heterogeneous Data for Question Answering", *Proceeding of the 7th International Workshop on Applications of Natual Language to Information System (NLDB'2002)*

Klas,W.; Fankhauser,p.; Muth, P.; Rakow,T.; and Neuhold, E.J. (1995), "Database Integration using the Open Object Oriented Database System VODAK", In *Elmagarmid, Ahmed and Bukhres, Omran, ED., Object-Oriented Multidatabases, Prentice Hall*

Lau, Henry C.W., and Wong, Eric T.T. (2001), "Partner Selection and Information Infrastructure of a Virtual Enterprise Network", *International Journal of Computer Integrated Manufacturing*, Vol. 14, No. 2, p186-193

Liang, Ting-Peng; Shaw, Michael J.P.; Wei, Chih-Ping (1999), "A Framework for Managing Web Information: Current Research and Future Directions", in *Proceedings of the 32nd Hawaii International conference on System Sciences*

Nijenhuis, Wim; Jonker, Willem; and Grefen, Paul (1997), "Supporting Telecom Business Processes by Means of Workflow Management and Federated Databases", in *the Proceedings of the Association for Infromation Systems 1997 Americas Conference, Indianapolis, Indiana, August 15-17*

Ozsu, M.Tamer, and Valduriez, Patrick (1996), "Distributed and Parallel Database Systems", ACM Computing Surveys, Vol. 28, No. 1, pp. 125-128

Panti, Maurizio; Penserini, Loris; and Spalazzi, Luca (2002), "A Pure P2P Approach to Information Integration", *in the Proceedings of the 28th Internal Conference on Very Large Databases (VLDB 2002), HK, China, August 20-23*

Papakonstantinou, Yannis; Gupta, Ashish; Garcia-Molina, Hector; and UllmanA, Jeffrey (1995), "Query Translation Scheme for Rapid Implementation of Wrappers", in the *Proceedings of 4th International Conference on Deductive and Object-Oriented Databases*

Perera, S; Harrison, G; Poole, M (1997), "Customer-focused Manufacturing Strategy and the Use of Operations-based Non-financial Performance Measures: A Research Note", *Accounting, Organizations and Society*, Vol. 22, No. 6, pp. 557-572

Pitoura, Evaggelia; Bukhres, Omran; and Elmagarmid, Ahmd (1995), "Object Orientation in Multidatabase System", *ACM Computing Surveys*, Vol. 27, No. 2, pp. 141-195

Prasad, S., and Rajaraman, Anand (1998), "Virtual Database Technology, XML, and the Evolution of the Web", *Data Engineering* Vol. 21, No. 2, pp. 48-52

Rajaraman, Anand (1998), "Virtual Database Technology: Transforming the Internet into a Database", *IEEE Internet Computing*, 1089-7801, pp. 55-58

Sadanandan, P,, and Chakravarthy S. (Eds.) (1994), ``Advances in Data Management '94", Tata McGraw Hill Publications

Sheth, Amit P. and Larson, James A. (1990), "Federated Database Systems for Managing Distributed, heterogeneous, and autonomous databases", *ACM Computing Surveys*, Vol. 22, No. 3, pp. 183 – 236

Stonebraker, Michael; Aoki, Paul M.; Pfeffer, Avi; Sah, Adam; Sidell, Jeff; Staelin, Carl; and Yu, Andrew (1996), "MARIPOSA: A Wide-Area distributed Database System", *VLDB Journal: Very Large DataBases*, Vol. 5, No. 1, pp. 48-63

Thiran, Ph.; Chougrani, A.; Hick, J-M.; and Hainaut, J-L. (1999), "Generation of Conceptual Wrappers for legacy Databases", in *Proceedings of DEXA'99, 10th International Conference and Workshop on Database and Expert Systems Applications, Florence, Lecture Notes in Computer Science – Springer-Verlag*

Thiran, Ph.; Chougrani, A.; Hainaut, J-L.; and Hick, J-M. (2000), "CASE Support for the Development of Federated Information Systems", in *Proceedings of EFIS'2000, 3rd International Workshop on Engineering Federated Information Systems, Dublin*

Thiran, Ph.; Hainaut, J-L.; Bodart, S.; Deflorenne, A.; and Hick, J-M. (1998), "Interoperation of Independent, Heterogeneous and Distributed Databases. Methodology and CASE Support: the InterDB Approach", in *Proceedings of CoopIS-98*

Wache, Holger; and Stuckenschmidt, Heiner (2001), "Practical Context Transformation for Information System Interoperability", *LectureNotes in Computer Science*,Vol. 2116, p367

Wadsack, Jorg P.; Niere, Jorg, Giese, Holger, Jahnke, Jens H. (2002), "Towards Data Depency Detection in Web Information Systems", *Database Maintenance and Reengineering Workshop (DBMR'2002)*

Wiederhold, Gio (1995), "Value-added Mediation in Large-Scale Information systems", Meersman (ed)*: Database Application Semantics,* Chapman and Hall

Wooldridge, Michael and Nicholas R. Jennings (1995), "Agent Theories, Architectures, and Languages: a Survey," in Wooldridge and Jennings Eds., *Intelligent Agents*, Berlin: Springer-Verlag, 1-22