

# FLEXIBLE DISTRIBUTED DATABASE MANAGEMENT WITH AgentTeam\*

Bora □ Kumova

Dokuz Eyl, l University, Dept. of Comp. Eng., 35100 İzmir, Turkey; kumova@cs.deu.edu.tr

## ABSTRACT

With respect to data format and data consistency, the spectrum of DDBM ranges from management of relational data in distributed database management systems (DDBMS) to the management of non-relational data in information retrieval systems. On the other, in many networked application domains users need to access any type of data, preferably with a single tool. However, DDBM is still a challenging research area that involves bridging syntactic and semantic heterogeneity of data as well as of functionality, especially in the bottom-up design of a new DDB. Since, existing DDBM systems were usually built with a focus on the implementation of some dedicated protocols for DDBM, they are inflexible for major modifications or exchange of the protocols. In addition, their software architecture usually does not comply with well-known design paradigms, which could facilitate the maintenance of the software system. We present an agent-based approach for flexible DDBM, where independent DDBM protocols are modular exchangeable for test purposes. This capability of the DDBMS provides for flexibility against protocol heterogeneity and enables the DDBMS to communicate with new DBMSs to be included, by combining or adapting its protocols at run time. For instance, different DDB consistency levels are possible for different DDBs with the same DDBMS. In this work, the design of DDBM in form of a multi-agent system of the AgentTeam framework is discussed, particularly the general query processing scheme. Furthermore, the CourseMan prototype is described, which is implemented in form of a test-bed that provides a homogeneous environment for testing different DDBM protocols, upon heterogeneous DBMSs.

**KEYWORDS.** Databases; artificial intelligence and expert systems; parallel and distributed processing

## 1. Introduction

Two principal organization types of relational data over the network are differentiated in the literature: top-down designed DDB and bottom-up designed multi DBs. In both cases, the major problem is the inter-operability between different types of DBMS, which is caused by syntactic and semantic heterogeneity in data models and in access models. For instance, relational data and objects [5], [19], different conceptual schema representations [11], [9], guaranteed and relaxed DB integrity [19], [24], and different DB tools need to be homogenized in a DDBMS. In order to cope with various types of heterogeneity, successful solutions

should attack the problem from two angles: algorithmic and software design oriented. The former is necessary in any case, whether a software design is used or not. The latter is used to facilitate the former. The most promising and general-purpose software design paradigm currently is object-orientation. Therefore, object-oriented solutions have been proposed for DB inter-operability [22], [4]. However, object-orientation alone is insufficient for designing intelligent behaviour. The most promising design paradigm for intelligent behaviour of a distributed system is the knowledge-based agent approach. Research in multi-agent systems (MAS) has reached a maturity that has already enabled its successful design, implementation, and commercialization, mainly in the information retrieval domain [7], [1], [2], [20], [25], [18].

Since many solutions to problems of relational DDBM require NP, heuristics are necessary to decide on non-deterministic functionality. Where a suitable representation form for non-deterministic functionality is the knowledge-based approach. Therefore, we have designed the AgentTeam framework for flexible DDBM in form of an intelligent MAS and implemented the CourseMan prototype in form of a test-bed [12].

A further concern of the prototype is exploratory testing of protocols for DDBM. Most implementations of DDBMSs are devoted to the implementation and test of some dedicated protocols for DDBM. For instance, Mariposa [24] was initially designed to test economy-based data exchange and scalability to a large number of co-operating sites. Nevertheless, it does not provide a test environment for different protocols. Our goal with CourseMan was to implement a test-bed for DDBM, flexible enough to provide for modular exchangeability of different protocols, where these themselves may introduce the system relaxed DDBM functionality.

In this work, the flexible DDBM of the AgentTeam framework is discussed, with an emphasis on the query processing scheme, the test-bed environment of the CourseMan prototype is introduced, and some aspects of the agent-approach are discussed. The paper concludes with some final remarks.

## 2. DDBM in AgentTeam

In the design of the AgentTeam framework, major problem areas of DDBM have been considered and some principal software design issues chosen. The framework consists of several models, of which the following are introduced here briefly: DDB model, DDBM model, multi-agent model, and the distributed query-processing model.

---

\* This paper was published at IASTED-AI'00

## Problem Areas of DDBM in AgentTeam

We have adopted following commonly known problems of DDBM as requirements for the DDBM model of AgentTeam. They have influenced design of the AgentTeam framework.

*Physical DB Design:* The physical design of relational DBs involves decision on the columns to be indexed for each relation. However, index selection is known to be NP-complete and therefore requires heuristics to be solved [3].

*DDB Design:* In the top-down design of a DDB, two basic alternatives for placing data are considered: fragmentation and replication. The design goal of minimizing the combined cost of storing the DB, processing transactions against it, and communication is NP-hard. Therefore, proposed solutions are based on heuristics [21].

*Distributed Query Processing:* It deals with designing algorithms that analyze queries and convert them into a series of data manipulation operations. The problem is how to decide on a strategy for executing each query over the network in the most cost-effective way. Factors to be considered are distribution of data, communication cost, and lack of sufficient locally available information [21]. Distributed query optimization is known to be NP-hard by nature and suggested heuristic solutions are inherently non-deterministic. Related discussion can be found in [26], [8].

*Distributed Directory Management:* For simplicity and in order to preserve autonomy this issue is left to the local DBMSs.

*Distributed Concurrency Control:* Concurrency control involves the synchronization of processes to the DDB, such that the integrity of the DB is maintained. The goal is to achieve mutual consistent values of multiple copies of data items. Solution alternatives are various alternative combinations of pessimistic/optimistic synchronization with locking/time-stamping or economy-based approaches [24].

*Distributed Deadlock Management:* Local wait for graphs need to be maintain as well as global wait for graphs.

*Heterogeneity:* We differentiate between two essential forms of heterogeneity in general: syntactic and semantic heterogeneity in data as well as in functionality. Heterogeneity directly related to DBM can occur in data models and data access languages.

Especially, heuristic solutions tend to result in non-deterministic functionality, which requires decision processes to terminate a related evaluation. A knowledge-based system is a suitable approach for implementing solutions for NP-hard and NP-complete problems within one system. This is due to the possibility to direct the search for a solution by dynamically adaptable heuristics.

## Principle Design Issues for AgentTeam

For our purposes, we define flexible DDBM with following major system properties: test-bed for DDBM, modular exchangeable and combined test of different protocols, and support for relaxed DDBM. In order to satisfy these properties, from the perspective of software engineering, following issues have guided the design of the framework:

*Distribution:* In order to provide a suitable environment for testing various types of protocols, the DDBMS itself should inherently be a distributed system. Accordingly, data as well as functionality will be distributed.

*Object-orientation:* Object-orientation is necessary to facilitate reuse, modularity, and scalability of software components inside a system. Furthermore, sub-typing and super-typing are the most powerful techniques to cope with heterogeneity, to facilitate DB interoperability, and system transparency [21]. Accordingly, the AgentTeam software architecture have been designed inherently object-oriented [Kumova 00b], [Kumova 00c].

*Agent-based Approach:* The agent approach is a software design paradigm that includes practical artificial intelligence (AI) techniques. We have adopted this approach, since the components of the DDBMS should be relative autonomous, which is an important property of intelligent behaviour and can provide the required system flexibility.

*Knowledge Representation:* The most important design goal for AgentTeam was to find a knowledge representation form flexible enough to represent heterogeneous information. We have decided to use semantic nets, since they are powerful enough to represent any data structure. Semantic nets are the only data structure of the knowledge base of an AgentTeam agent. LISP Processing (LISP)-like code can be attached to a node of a semantic net, in order to represent a possible operation on that data item. Semantic nets are already used for the representation of heterogeneous data, for instance in Object Exchange Management (OEM) [22]. OEM is implemented in the information retrieval systems The Stanford-IBM Management of Multiple Information Sources (Tsimmis) [6] and Lightweight Object REpository for semi-structured data (Lore) [17].

*Client/server Model:* Communication relationships between the autonomous components should be designed according the client/server model, where synchronous and asynchronous modi should be supported.

*Flexible DDBM:* The system should allow for modular exchangeability and combination of different protocols inside a test environment. It should support the implementation of relaxed DDBM functionality.

## DDB Model of AgentTeam

Basic idea of the DDB model is to utilize shared DBs dispersed over an unlimited number of heterogeneous DBMSs at different sites. The logical and physical relationships between DBs, DDBs, and DBMSs, inside the DDBMs of AgentTeam are depicted in (Figure 1)

with different visualizations. The cardinalities between the logical system components represent unrestricted scalability. Some important relationships are:

- The number of all replicated DBs of the whole system is defined by  $(pCq \wedge DBp=DBq)$ .
- $x$  and  $y$  are the number of DBs in  $DBMS_i$ , which are not included in  $AgentTeam$ .
- $t_i$  is the number of DBs included in  $AgentTeam$  and locally managed by  $DBMS_i$ .
- $k_i$  is the number of DBs that constitute  $DDB_i$ .
- $n$  is the number of  $DDBs$  managed by  $AgentTeam$ .

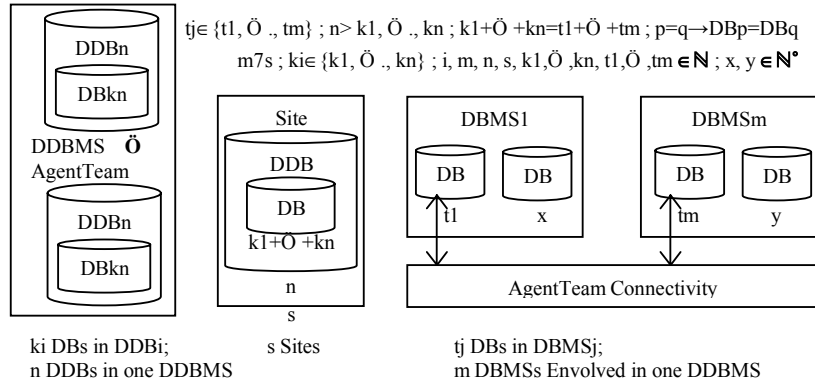


Figure 1: Relationships Between Data Components Managed by AgentTeam

### DDBM Model of AgentTeam

Following issues are supported in  $AgentTeam$ : Multiple  $DDBs$  can be managed simultaneously in the  $DDBMS$  (Figure 1). The autonomy of each  $DBMS$  is guaranteed, since the local  $DBMSs$  are not aware of  $AgentTeam$ . However, data distribution and data consistency is managed by  $AgentTeam$ . Transaction management is implemented at client site, concurrency control and scheduling are implemented at server site.

Protocols for following  $DDBM$  functionality are provided in their basic form: query processing and optimization, concurrency control, replication management, and fragmentation management.

### Multi-agent Model of AgentTeam

Four agent types are employed in  $AgentTeam$  (Figure 2) [15]:

- A user agent implements a domain-specific user interface. It invokes for each user transaction one task agent.
- A task agent implements transaction management and distributed query processing.
- Resource agents implement broker functionality that is utilized by task agents to find  $DB$  agents.
- $DB$  agents implement concurrency control and recovery management.

The communication relationships between the agent types are discussed elsewhere [13], [14], as the communication model of the  $MAS$ . All mentioned  $DDBM$  protocols are implemented upon this model. Further discussion of the responsibilities of each agent type, with respect to distributed query processing is given below.

The following is a sample scenario for schema integration. A user determines a  $DDB$  or makes structural modifications. The related user agent invokes a task agent, which informs the involved  $DB$  agents. The  $DB$  agents negotiate, to find one suitable  $DB$  agent, which has to perform schema integration. This  $DB$  agent co-ordinates the schema integration. Finally, the global schema is replicated to all involved  $DB$  agents.

### Distributed Query Processing in AgentTeam

From the above problem areas we will discuss here only the distributed query processing scheme of  $AgentTeam$  as it is implemented in  $CourseMan$  (Figure 3). In the following overview, the related process steps are described briefly.

*Java Graphical User Interface (GUI)*: Transforms the textual representation of a query into a simplified LISP representation.

*Parsing*: The parser accepts a LISP representation of a query, transforms it into a parse tree, and then passes the parse tree to the pre-processor.

*Global Schema Refresh*: Before the pre-processing the current global schema is refreshed. If it is not up-to-date, then it is copied from one of the related  $DB$  agents.

*Query Pre-processing*: The pre-processor analyzes the parse tree and adds further information extracted from the global schema, such as location of data and cost for functionality.

*Query Plan Generation*: The enriched parse tree is analyzed by the query generator and alternative query execution plans are generated without considering the current availability of a  $DB$  agent.

*Query Optimization*: The optimizer sorts alternative plans according to their total costs and decides whether the use of indexes is feasible.

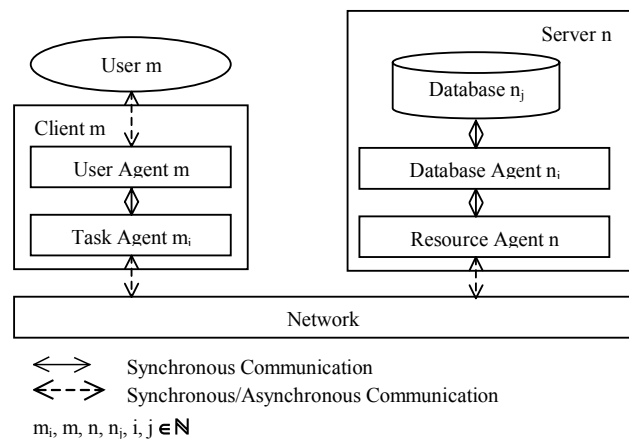


Figure 2: Communication Relationships between the Autonomous Entities

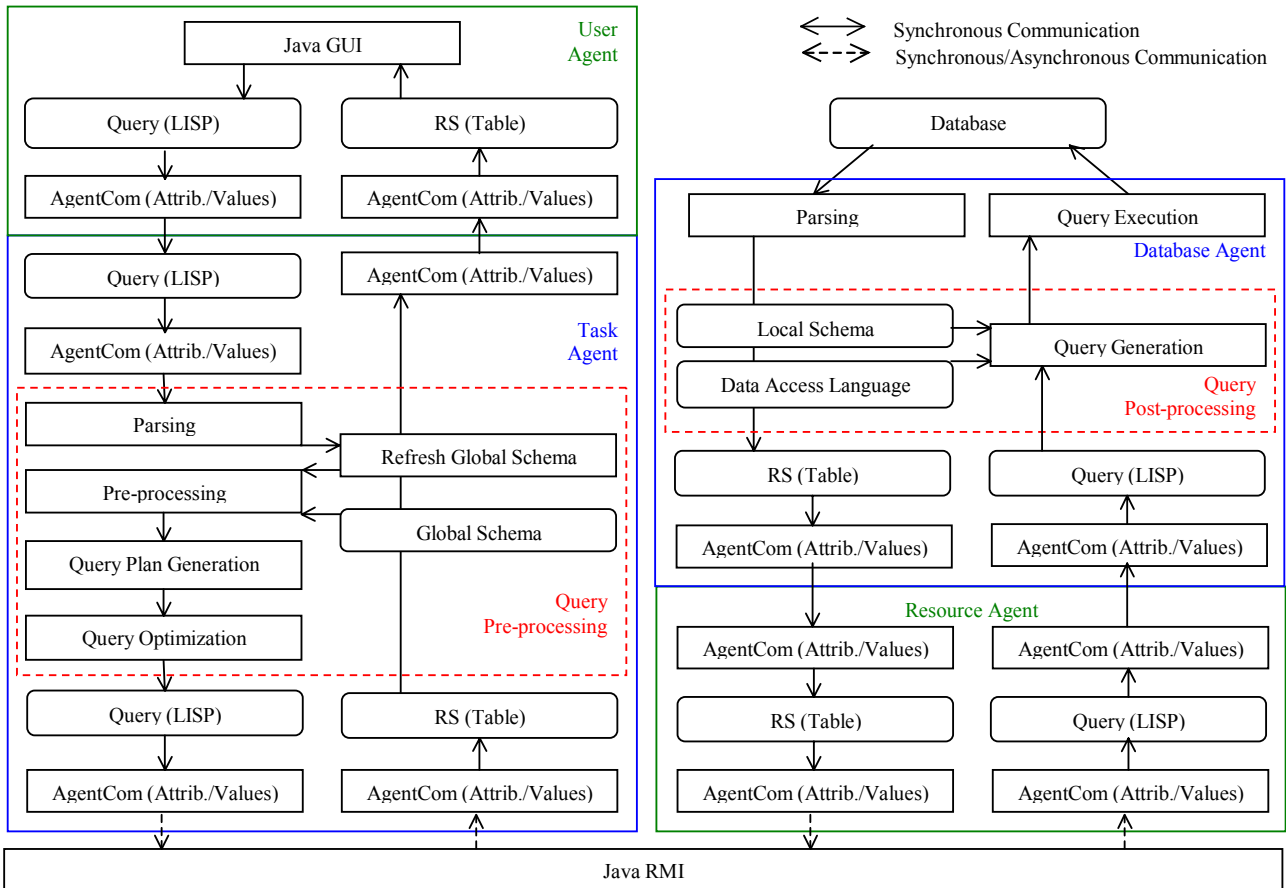


Figure 3: General Query Processing Scheme of CourseMan with Data Structures

*Query Post Processing:* The query in LISP representation is transformed into the data access language of the local DBMS. If necessary, the local schema can be utilized in this process.

### 3. CourseMan Prototype Implementation

Some of the ideas and models of the AgentTeam framework are implemented in the CourseMan prototype. It is a test-bed for testing the inter-operability of different protocols.

Most protocols for DDBM are extensions, originally designed for centralized DBM, and therefore usually need to be modified and tested to suit the distributed environment. Some important test issues are time constraints, parameter variations, harmonic functioning with other protocols, and reliability.

*Test-bed Features:* Due to the interpreter mode, the DDBMS is run-time re-configurable with respect to protocols and other system functionality. A DDB is constituted by the involved DBs and the number of protocols for its management. Usually, a set of different protocols to be tested is stored inside the DDBMS. Where task agents implement the client-site module and DB agents the server-site module of a specific protocol (Figure 4). If a user modifies the properties of a selected DDB, then the protocol parameters are adapted accordingly, and/or the protocol is exchanged, in order to fit the required DDBM functionality. Such a modification can be initiated by a task agent or by DB

agents. It is communicated and performed mutually with all agents that are involved to the management of the related DDB. A DDBM protocol is represented in form of objects, whereby the functionality is coded in a LISP-like syntax and stored in the agent's knowledge base [16].

*Result Set Handling:* Inside the address-space of the applications, a result set (RS) is passed between the modules always by a reference to the data block. The structure of an RS is transformed only once in total, from a rough table format into a simple table format. Since, the transformation is made on the property format but not on the data itself, the RS data is actually not moved. On the way from the DB to the Java GUI, a RS is copied only three times in total. First, from the DB into the server-site application, from there to the network, and finally to the client-site application.

*Performance Considerations:* In this solution no

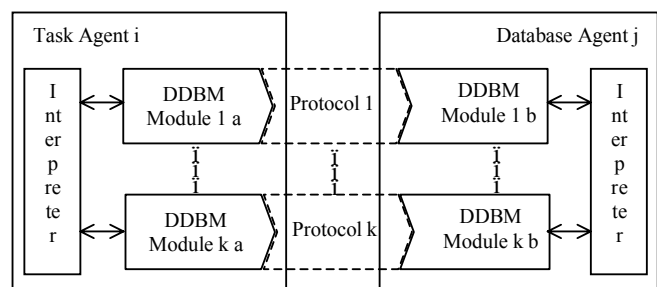


Figure 4: Generic Architecture for Modular Exchangeable Protocols

additional data movements are performed. Additional processing is required for the evaluation of LISP-like code in the interpreter mode, which is the operation mode of the whole system. Since, the processing of the code is independent from the data amount, the additional process time required for interpreted code is minor in case of large RSs. However, the implementation language Java [10] introduces considerable overhead in processing time, which is in general 10-to-1 of C in the average, since Java itself runs in interpreter mode.

*Scalability:* Since, the addresses of the DDBMS sites are maintained in lists and are stored by each agent, the system is scalable to a large number of involved sites, DBMSs, and DBs.

*System Configuration:* Currently, the prototype consists of four server sites, at which four different commercially available SQL DBMSs are running. All DBs are accessed via ODBC.

#### 4. Discussion

The application domain of the AgentTeam DDBM model are structured DBs, such as relational and object-oriented DBs, which are small in size, are networked and require flexible and distributed management.

For the implementation of non-deterministic functionality, such as optimization of distributed queries [8], [26], the knowledge-based system proved to be a suitable approach, due to the run-time re-configurable capability.

The agents of AgentTeam communicate in the agent communication language AgentCom [16], which provides independence from data structures of local DBMSs

Another feature of AgentTeam is that the transition from DDBM to multi DBM or multi DDBM are floating, which can be manipulated through related system configurations.

An early project on flexible DDBMS was Mariposa [24], [23]. This system was restricted to the test and experimentation of the economy-based management of a single DDB. The result was that relaxed global synchronisation can increase local DBMS autonomy and enable high system scalability.

Another approach for DBMS inter-operability is based on object [5]. Here however, the objects are not provided with intelligent behaviour.

#### 5. Conclusion

We have discussed AgentTeam, which is a framework for flexible DDBM in form of a MAS. CourseMan is a prototype implementation in form of a test-bed. It provides a homogeneous environment for testing different DDBM protocols on heterogeneous DBMSs. It runs on heterogeneous platforms, at run-time it allows changing the parameters of current protocols, as well as exchanging a protocol wholly. This capability provides the system with flexibility against various types of DB heterogeneity.

Currently, the system is extended by a pure object-oriented DBMS. Besides, some of the functionality is optimized, in order to improve the response times.

#### References

Referenced URLs were valid by February 10, 2000.

- [1] Boles, Dietrich; Dreger, Markus; Groffjohann, Kai; 1996; iKonzeption eines Informationsvermittlungssystems f, r heterogene, verteilte Informationsquellen im Interneti;
- [2] Bowman, C. Mic; Danzig, Peter B.; Hardy, Darren R.; 1995; iHarvest: A Scalable, Customizable Discovery and Access Systemi; TR CU-CS-732-94; University of Colorado; Boulder
- [3] Do5aÅ Asuman; 1991; iAn Expert System for Physical Design of Relational Databasesi; *Expert Systems and their Microcomputer Applications*; Ankara
- [4] Do5aÅ Asuman; Dengi, Cevdet; ÷zsu, M.Tamer; 1998; iBuilding Interoperable Databases on Distributed Object Management Platformsi; *Communications of the ACM*
- [5] Fang, Douglas; Hammer, Joachim; McLeod, Dennis; Si, Antonio; 1999; iRemote-Exchange: An Approach to Control Sharing among Autonomous, Heterogeneous Database Systemsi; University of Southern California
- [6] Garcia-Molina, Hector; Hammer, Joachim; Ireland, Kelly; Papakonstantinou, Yannis; Ullman, Jeffrey; Widom, Jennifer; 1995; iIntegrating and Accessing Heterogeneous Information Sources in TSIMMISi; *AAAI Symposium on Information Gathering*; Stanford
- [7] Genesereth, Michael R.; Keller, Arthur M.; Duschka, Oliver M.; 1997; iInfomaster: An Information Integration Systemi; *ACM SIGMOD*
- [8] Groöelj, Bojan; Malluhi, Qutaibah M.; 1995; iCombinatorial Optimization of Distributed Queriesi; i; *IEEE Transactions on Knowledge and Data Engineering*, Vol.7
- [9] Hammer, Joachim; 1994; iResolving Semantic Heterogeneity in a Federation of Autonomous, Heterogeneous Database Systemsi; PhD thesis; University of Southern California
- [10] JDK; 1999; "JDK 1.2 Documentation"; Sun Microsystems Inc., Palo Alto
- [11] Kashyap, Vipul; Sheth, Amit; 1995; iSemantic and Schematic Similarities between Database Objects: A Context-based approachi; *The VLDB Journal*
- [12] Kumova, Bora □; 1998; iSystem Specification for an Example Distributed Databasei; MSc; D.E.U., Dept. of Comp. Eng.
- [13] Kumova, Bora □; Kut, Alp; 1998; iA Communication Model for Distributed Agentsi, *ISCI98, Concurrent Systems Engineering Series*, Volume 53, IOS Press, Amsterdam
- [14] Kumova, Bora □; Kut, Alp; 1999; iAgent-based System Cooperationi; *PDP99*; Madeira; <http://cs.deu.edu.tr/~kumova>
- [15] Kumova, Bora □; 1999; iThe Agent Model of AgentTeami; *TAINNi99*; Istanbul; <http://cs.deu.edu.tr/~kumova>
- [16] Kumova, Bora □; 2000; iThe Agent Communication Language AgentCom and its Semanticsi; TR; <http://cs.deu.edu.tr/~kumova>
- [17] McHugh, Jason; Abiteboul, Serge; Goldman, Roy; Quass, Dallan; Widom, Jennifer; iLore: Adatabase Management System for Semistructured Datai; *SIGMOD97*; Sep.
- [18] Martin, David; Cheyer, Adasm J.; Moran, Douglas B.; 1999; iThe Open Agent Architecture: A Framework for Building Distributed Software Systemsi; *Applied Artificial Intelligence*, vol. 13, pp. 91-128
- [19] Mehrotra, Sharad; Rastogi, Rajeev; Korth, Henry F.; Silberschatz, Abraham; 1991; iMaintaining Database Consistency in Heterogeneous Distributed Database Systemsi; technical report; University of Texas at Austin
- [20] Odubiyi, Judé B.; Kocur, David J.; Weinstein, Stuart M.; Wakin, Nagi; Srivastava, Sadanand; Gokey, Chris; Graham, JoAnna; 1997; iSAIRE ñ A Scalable Agent-based Information Retrieval Enginei; *Autonomous Agents97*, Marina Del Ray
- [21] ÷zsu, M. Tamer; Valduriez, Patrick; 1999; iPrinciples of Distributed Database Systemsi, Prentice Hall, Englewood Cliffs
- [22] Papakonstantinou, Yannis; Garcia-Molina, Hector; Widom, Jennifer; 1995; iObject Exchange Across Heterogeneous Information Sourcesi; *IEEE International Conference on Data Engineering*; Taipei
- [23] Sidell, Jeff; Aoki, Paul M.; Barr, Sanford; Sah, Adam; Staelin, Carl; Stonebraker, Michael; Yu, Andrew; 1996; iData Replication in Mariposa; *12th International Conference on Data Engineering*; New Orleans

- [24] Stonebraker, Michael; Aoki, Paul M.; Litwin, Witold; Pfeffer, Avi; Sah, Adam; Sidell, Jeff; Staelin, Carl; Yu, Andrew; 1996; 'Mariposa: a wide-area distributed database system'; *The VLDB Journal*, Vol. 5, pp. 48-63, Springer-Verlag, Berlin
- [25] Sycara, Katia; Decker, Keith; Pannu, Ananddeep; Williamson, Mike; Zeng, Dajun; 1997; 'Distributed Intelligent Agents'; *IEEE Expert*, Dec 96, <http://www.cs.cmu.edu/~softagents>
- [26] Wang, Chihping; Chen, Ming-Syan; 1996; 'On the Complexity of Distributed Query Optimization'; *IEEE Transactions on Knowledge and Data Engineering*, Vol.8