マルチメディア通信と分散処理　57－16
(1992. 9. 25)

協調的データベースシステムについて

矢羽田　千哲、濱田　賢、滝沢　誠
東京電機大学理工学部経営工学科
埼玉県比企郡鳩山町石坂

あらまし　データベースシステムは、通信網によって結合された自律的なデータベースシステムから構成される。利用者に対して、システムでは、相互運用性を提供しなければならない。本論文において、相互運用性とは、利用者が多データベースシステムの視野を動的に定義できることとする。従来の分散型データベースシステムでは、異種性と分散性を意識せずに複数の多データベースシステムを利用させることが目的である。安全性の制約によって、演算の実行に対する権限付与がなされているならば、各データベースシステムは、利用者からの演算が実行できることを仮定する。しかしながら、各データベースシステムが演算を実行しない場合や、演算の実行を受けつけても、負荷が高いといった各データベースシステムの内部状況により、応答を迅速に返せない場合がある。このように、各データベースシステムは、自律性を持っている。データベースシステムの自律性とは、データベースシステムが演算をいつ、どのように実行するかを決定できることである。本論文では、自律的なデータベースシステムにおける相互運用性をどのように提供するかについて論ずる。

和文キーワード　データベース、自律性、分散システム、相互運用性

Cooperating Database System

Chiaki Yahata, Satoshi Hamada, and Makoto Takizawa
Dept. of Computers and Systems Engineering
Tokyo Denki University
Ishizaka, Hatoyama, Hiki−gun
Saitama 350−03 JAPAN
e−mail { chii, hama, taki }@takilab. k. dendai. ac. jp

Abstract　A cooperating database system includes autonomoys database systems interconnected by communication networks. The system has to provide the interoperability among the database systems for users. In this paper, the interoperability means that users can define dynamically their views on multiple database systems.The classical distributed database systems aim at providing services where users can use multiple database systems without being conscious of their heterogeneity and distribution. Further, it is assumed that each database system can accept operations from users if the operations are authorized by the security constraint. However, each database system may not take the operations or not reply them quickly due to the internal situation like overloaded one even if the operations can be accepted. Thus, each database system has some autonomy, i.e. can decide what and how it can do for each operation. In this paper, we discuss how to provide the inter operability on autonomous database systems.

英文 key words　database, autonomy, distributed system, interoperability

1 Introduction

Current information systems include various kinds of database systems interconnected by communication networks [8, 14]. The system has to provide users with the interoperability among multiple database systems. The classical distributed database systems [6, 9, 10, 11, 16, 20] tried to provide service by which users can use multiple database systems without being conscious of their heterogeneity and distribution. There are two kinds of distributed database systems, i.e. integrated and multi-database systems. In the integrated ones, one global integrated schema is provided for users on multiple database systems. In the multi-database systems [12], users can define their users on multiple databases. In the distributed database systems, it is assumed that each database system takes operations if the operations are acceptable from the integrity and security point of view. However, each database system might not take the operations or not reply them quickly due to the internal situation, i.e. it is overloaded. Thus, each database system has some autonomy [15], i.e. each database system can decide what and how it can do for each operation. On the other hand, the security is concerned with which part of the data structure and operations which users can access. In the conventional distributed database systems, it has not been discussed how to take into account the dynamic autonomy. Recently, the autonomy is discussed in the distributed artificial intelligence [7].

The second point is that the system configuration is dynamically changed. For example, database systems may join and leave the system. Some database system may change the data structure and the operations. That is, users may not know everything about the system state, i.e. which and how database system can be used now. Users have to identify which database systems can be accepted to get information required and can be manipulated before issuing the operations.

A cooperating database system (CDBS) is a system which is composed of multiple autonomous nodes interconnected by communication networks. The nodes are database systems or user clients. In this paper, the interoperability means that users can define dynamically their views on multiple autonomous database systems. In this paper, we would like to present how to provide the interoperability on the autonomous database systems based on the cooperation among them and users.

In section 2, we present the system model of the distributed database system. In section 3, we discuss acquaintances of the nodes. In section 4, we present how to negotiate among the database systems to answer the requests from users. In section 5, we present how to adopt the change of the system.

2 System Model

We present a model of the cooperating database system (CKBS.)

2.1 System configuration

A cooperating database system (CKBS) is composed of multiple nodes which are in-

terconnected by communication networks. Each node has a type, which defines the possible states and the operations for manipulating the state. There are two kinds of nodes, i.e. ***active*** and ***passive*** ones. The active node can not only take the operations from users or another nodes but also can issue the operations to another node. On the other hand, the passive node can only take the operations from another node but cannot issue operations to another node. The conventional server system like the file server is an example of the passive node. Suppose that a passive node a accepts an operation ***op*** from another b. If a cannot get any answer of ***op*** from the state, a sends the negative reply back to b. On the other hand, if the node b is active, a may issue operations to another nodes to get the answer of ***op***.

Each node is either a client or a database system. The client is an interface system among users and the database systems. The client is an active node. i.e. it can issue the operation to another nodes. It takes an operation ***op*** from the users, decomposes it to suboperations issues the suboperations to the nodes to answer the operation ***op***, and composes the answer for ***op*** from the answers obtained by the nodes to which the operations were issued. It is typically realized in the user's workstation. A database system is a system which provides users with some data model. There are two kinds of the database systems. The first one is a passive database system which can only take the operations from another nodes. The other is an active database system, which is a combination of the client and the passive database system. The conventional database system is a passive one.

2.2 Heterogeneity

In order to manipulate data in multiple database systems, we have to consider two points, i.e. ***heterogeneity*** and ***autonomy*** of the database systems. Two database systems are ***syntactically heterogeneous*** if they provide different types of data models, e.g. one is relational [2] and the other is network-typed [1]. There have been many discussions on the syntactical heterogeneity [16] already. In this paper, every database system is assumed to be relational. In a case that the database system is not relational, we can suppose that it provides a relational interface system [16].

The database systems have ***semantically heterogeneous*** data. For example, there exists some name conflict, i.e. the same name in different database systems may denote different objects, and different names may denote the same object. It is still problem to solve the semantic heterogeneity. In this paper, every database system is assumed to be semantically homogenized. That is, the same name in every database system denotes the same object. In this paper, the database is assumed to be characterized by a set of ***terms***, i.e. keywords. For example, a database on ***Japan travel*** includes terms ***Japan***, ***travel***, ***train***, ... on the traveling in Japan.

2.3 Autonomy

In order to integrate multiple database systems, we have to take into account the autonomy. A database system D is

autonomous if D can decide dynamically by itself (1) which data in D can be used by which node, and (2) which operations in D can be executed by which node. There are various levels of the autonomy. D is completely autonomous if D can decide everything about it by itself. D is partially autonomous iff D is autonomous but not completely autonomous.

2.4 Layer Structure

Each database system D provides layers for users. The lowest one is a local (L) model which corresponds to the conceptual layer [20] of D. It is based on various model, e.g. relational model, network model. The second layer is a local conceptual (LC) model which represents the local model in terms of a common type of a data model. At this level, each database system is viewed to be syntactically homogenized. Further, we assume that each LC model is equivalent to the local model and every operation on the LC model can be translated to the operations on the local model. In this paper, we assume that each LC model is based on the relational model. Also, we assume that there is no semantic heterogeneity among the LC models. That is, each term denotes the same object in every database system. At the LC model layer, every database system can be viewed to be homogeneous. That is, users can use it as it were relational.

An export (EP) model of the database system D is a subset of the LC model, which includes only data and operations which can be used by another database system. The EP model is defined dynamically through negotiation among the database system D and users which would like to use D. In this sense, each database system provides autonomy dynamically by providing EP models.

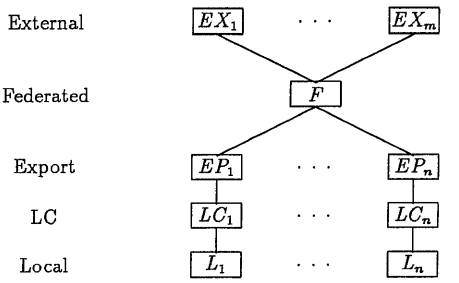

Figure 1: Five-layer Structure

If the user would like to get some information from the cooperating database systems (CKBS), it defines a federated model from the EP models. The EP model EP_i is defined for the user through the negotiation among the user and the local database system D_i. In this paper, the interoperability means that users can dynamically define the federated model F. For each application, one external model EX_j is defined on the federated model F.

3 Acquaintances

Each active node, i.e. user and active database system would like to get some information on what data another node has. Nevertheless, the user may not know which node has the information and how reliable the information is even if the user has it. A node a is said to ***directly*** know of a term t if a has data denoted by t in a. a is said to know of t if a directly knows of t or knows another node b which knows of t. For each node a, let $K(a)$ be a set of terms

which a knows, $DK(a)$ be a set of terms which a directly knows of, and $ID(a)$ be $K(a) - DK(a)$. In a, for every term t, $node(a,t)$ be a set of nodes which a knows knows of t. Suppose that some node ask a to get data denoted by t. a can derive data from the database if t is in $DK(a)$. a can ask another node in $node(a,t)$ to get the data if t is not in $DK(a)$ but t is in $K(a)$. Otherwise, a cannot reply it. $K(a)$ represents what information a knows exists in the system. $K(a)$ is structured by the generalization and aggregation. A node b is said to be an ***acquaintance*** of a (written as $a \rightarrow b$) iff a knows of some term t which b knows, i.e. b is in $node(a,t)$. Let $AQ(a)$ be a set of acquaintances of a, i.e. $\{\ b \mid a \rightarrow b\ \}$. There are kinds of the acquaintances of a on the reliance. Since each node has the autonomy, even if a thinks that b is the acquaintance on a term t, b may not be the acquaintance now. Each node may change the type, e.g. it cannot respond some operation sometime, and it changes the scheme of the data structure. Hence, even if a knows of b, a might not have the up-to-date information on b. The node b is a trustworthy acquaintance of a iff a could know about the current state of b. For example, a trustworthy acquaintance b informs a of the change of b if b has some change. Let $TA(a)$ be a set of trustworthy acquaintances of a.

Let us consider an example as shown in Fig.2. There are clients A, B, C which know of ***Asia***, ***Europe***, and ***Travel***, respectively. The terms which nodes know of are structured as shown in Fig.2. There are three database systems I, J, and K which have data on ***Japan***, ***Europe***, and ***World*** travel, respectively. A and C are trustworthy acquaintance of a user client a. Dashed lines represent indirect know relations.

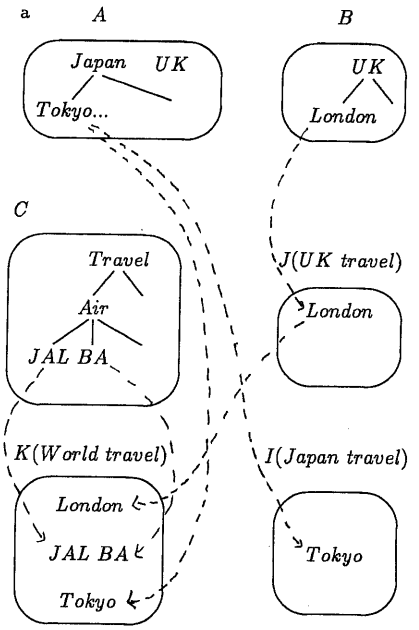

Figure 2: Example

4 Negotiation and Planning

User's queries are supposed to be composed of terms and operations on the terms. In this paper, only the retrieval operations are considered. First, a user u does not have enough information on database systems. u has only terms which u would like to get. Let $Q(u)$ be a set of terms included in the query of u. ***Negotiation*** is a process among the nodes to get information on which nodes have the information and how much to pay for getting the information. ***Planning*** is a

process to decide how to get the answers from the nodes based on the information obtained by the negotiation.

First, u looks for a trustworthy acquaintance of u which knows of some terms in $Q(u)$. If u finds the nodes a_1, ..., a_m in $TA(u)$ which have terms in $Q(u)$, u can issue queries to them. u selects the nodes b_1, ..., b_k from a_1, ..., a_m which are expected to give the answers with the minimum cost based on the performance information. u issues the queries to b_1, ..., b_k. Here, let $TQ(u)$ be $\{ t \mid node(u,t) \subseteq \{ b_1, ..., b_k \}$ which is a set of terms in $Q(u)$ which can be answered from the acquaintance. In Fig.2, suppose that a user a would like to get the information of trip to ***Japan*** from ***London***. Here, $Q(a) = \{$ ***Travel***, ***Japan***, ***London*** $\}$. Since A and C are trustworthy acquaintances of a, a knows that A has some information on ***Japan*** and ***UK***, and C has travel information. a asks A about ***Japan*** and ***UK***. Since A does not know directly about ***UK***, B asks C to get information on ***UK***. C informs A that a node I has information on ***London***. Then, A informs a that information on ***Japan*** and ***UK*** can be obtained from J and I, respectively. Further, a can know that the information on ***airlines*** and ***Japan*** and ***UK*** can be obtained from a node K. a decides to send the query to K since the access cost is expected to be less than I and J, i.e. only one node K is accessed in this case.

Next, the answers have to be obtained for the terms in $UQ(u) = Q(u) - TQ(u)$. Since the untrustworthy acquaintance might not respond the queries, u has to confirm they can answer the queries. u asks the acquaintances which u knows have terms in $UQ(u)$ whether or not they know of the terms now. If so, u issues the queries to the acquaintances which really know of the terms.

Lastly, there are still terms which any acquaintance does not know of. In this case, u ask every non-acquaintance node if it knows of the terms. Here, we cannot send the queries to every node in the system due to the communication cost. Each node belongs to a cluster in our system. Every node can reliably broadcast messages to all the nodes in the cluster [17, 21, 22]. A set of nodes interconnected by a local area network [8] is an example of the cluster. In the cluster, nodes can ask every other node to give information if it knows. If no node in the cluster has the information, the node must get it from another clusters. If it cannot be decided which node knows of the terms, u gives up to answer the queries, and gives the partial answer to the user.

5 Learning Module

Through issuing the queries, the user u gets not only the answers but also makes the model of u more complete. The model of u is a generalization and aggregation structure of terms. Each term t has a set of the properties $< nd, tp, pf >$ which means a node nd knows of t in a reliance tp with the performance pf. Also, there exists some generalization or aggregation relation r among two terms t_1 and t_2. Through negotiation among the nodes, u gets not only terms but also relations among the terms. u constructs the generalization and aggregation hierarchy among

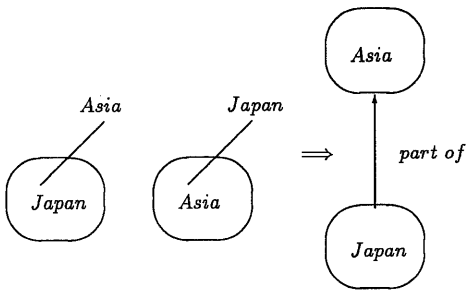

Figure 3: Learning Module

the terms by importing them from another nodes. In Fig.2, first, a does not know the relation among ***Japan*** and ***Asia***. After the negotiation, a gets the aggregation relation from ***Japan*** and ***Asia*** from the client A. Also, u changes the term structure. For example, if the trustworthy acquaintance cannot answer the queries which it had answered before, it is changed to be an untrustworthy one.

6 Concluding Remarks

In this paper, we have discussed what is the interoperability of the distributed information system. The interoperability has been defined to be the extendibility in this paper. In order to realize the interoperability, each node has some learning module (LM) to get information from another nodes. We have discussed how to get the information so as to minimize the communication cost.

References

[1] CODASYL, *"Data Description Language Journal of Development,"* Canadian Government Publishing Centre, 1973.

[2] Codd,E.F., *"A Relational Model of Data for Large Shared Data Bank,"* CACM, Vol.13, No.6, 1970, pp.337-387.

[3] Deen, S. M. et al.,*"The Architecture of a Distributed Database System - PRECI*,"* Computer Journal, Vol.28, No.3, 1985, PP.282-290.

[4] Banks, B. J., Deen, S. M., et al., *"Design and Implementation of DEAL,"* Proc. of the Working Conf. of Data and Knowledge Base Integration, Keele, 1989, pp.29-62.

[5] Deen, S. M., *"Cooperating Agents - A Database Aspective,"* Proc. of International Working Conf. on Cooperating Knowledge Based Systems, Keele, England, 1990.

[6] Heimbigner, D. and McLeod, D., *"A Federated Architecture for Information Management,"* ACM Trans. on Office Information Systems, Vol.3, No.3, 1985, pp.253-278.

[7] Huhns, M. N., *"Distributed Artificial Intelligence,"* Pittman, 1987.

[8] "IEEE Project 802 Local Network Standards-Draft," 1982.

[9] Proc. of the IEEE, Special Issue on Distributed Database System, 1987

[10] Landers, T. and Rosenberg, R., *"An Overview of Multibase,"* Distributed Databases (Schneider, H. -J. ed.), North-Holland, 1982, pp.153-184.

[11] Litwin, W. et al., *"SIRIUS Systems for Distributed Data Management,"* Distributed Data Bases (Schneider, H.-J. ed.), North-Holland, 1982, pp.311-366.

[12] Litwin, W. and Abdellatif, A., *"An Overview of the Multidatabase Manipulation Language MDSL,"* in [8], pp.621-632.

[13] Litwin, W., Mark, L., and Roussopoulos, N., *"Interoperability of Multiple Autonomous Databases,"* ACM Computing surveys, Vol.22, No.3, 1990, pp.265-293.

[14] "Data Processing - Open Systems Interconnection - Basic Reference Model," ISO 7498, 1980.

[15] Sheth, A. P. and Larson, J. A., *"Federated Database Systems for Managing Distributed, Heterogeneous, and Autonomous Databases,"* to appear in ACM Computing Surveys, 1990.

[16] Takizawa, M., *"Distributed Database System - JDDBS,"* JARECT Vol.7, Com puter Science and Technologies (Kitagawa, T., ed.), Ohmsha and North- Holland, 1983, pp.262-283.

[17] Takizawa, M. and Nakamura, A., *"Partially Ordering Broadcast (PO) Protocol,"* Proc. of the 9th IEEE Conf. on Computer Communications (INFOCOM), San Francisco, 1990, pp.357-364.

[18] Thomas, G., et al., *"Heterogeneous Database Systems for Production Use,"* ACM Computing Surveys, Vol.22, No.3, 1990, pp.237-266.

[19] Tsichritzis, D. and Klug, A., *"The ANSI/X3/SPARC DBMS Framework,"* Information Systems, Vol.3, No.4, 1978.

[20] Wilmes, P.F. et al., *"I wish I were over there: Distributed Execution Protocols for Data Definition in R*,"* Proc. of the ACM SIGMOD, 1983, pp.238-242.