HETEROGENEITY IN THE DISTRIBUTED DATABASE MANAGEMENT SYSTEM SIRIUS-DELTA

Arlette FERRIER, Christine STANGRET

INRIA, B.P. 105, 78153 Le Chesnay Cedex, France

Within the pilot project SIRIUS (ADI, INRIA), we have studied the problems set by the cooperation of heterogeneous components in a DDBMS.

Heterogeneity appears at many levels : computer, DBMS, data manipulation language, etc. The aims of a heterogeneous DDBMS are :

- to allow the users to manipulate the distributed database like a unique database and with the language they use to practice,
- to make possible the integration of an existing database in the DDBMS.

In the SIRIUS-DELTA heterogeneous prototype, computers as well as DBMSs and languages are different. We have designed a pivot system, that is to say a set of functions which have to be ensured by every system in the DDBMS. These functions are implemented using the existing services of the different systems.

We have studied a pivot model and a pivot language and implemented it on the DDBMS SIRIUS-DELTA and two other systems :

- PHLOX, a navigational DBMS for micro-computers,
- MRDS, a relational DBMS which runs on the MULTICS operating system implemented on a H.B. 68.

Introduction

In the last few years, much attention has been placed on the study of distributed database management systems. Implementations have already being made, using homogeneous networks of computers. A new problem is now merging : how to build heterogeneous DDBMSs, using existing database systems ?

In this paper, we first define the characteristics of a heterogeneous DDBMS (section 1). Then we present the heterogeneous DDBMS we are implementing. This prototype relies on the SIRIUS-DELTA homogeneous DDBMS and two local DBMSs : MRDS (running on a very large computer) and PHLOX (a DBMS for micro-computers). The basis of the system is a pivot system concept (section 2). The description of the distributed database schema refers to a pivot model which is the relational model. A pivot language, called PETAL, has been designed for the transfer of queries. On each user site, a query must be decomposed and translated into a set of PETAL sub-queries. On each site where data are stored, a PETAL sub-query is translated into the usual local manipulation language.

In section 3 and 4, we describe the different systems used in the heterogeneous SIRIUS-DELTA prototype and the interfaces we have implemented in order that they keep to the pivot concepts. An example of query processing is described in an appendix.

1. Heterogeneous distributed databases

A DDBMS relies on several databases managed by several systems running on several machines. In most systems implemented up to now, all components are homogeneous, that is to say that all the functions are realized in the same way on all the sites of the DDB (same computer and same software).

As soon as one of the functions is assumed in the whole DDBMS by components of different types, we can say that the system is heterogeneous for this function.

Many components can be heterogeneous in a DDBMS :

- the networks that interconnect the different sites (the system can use several local networks connected on a national network),
- the Data Description Languages (DDL) and the Data Manipulation Languages(DML),
- the DBMSs and each function they ensure : protection, synchronization, resources allocation, transactions management, ...,
- data models (relational, network, hierarchical) at the three levels [ANSI75] : external, conceptual and internal,
- the operating systems on which the DBMSs are running,
- the computers.

Various heterogeneous systems can be imagined : different computers with different operating systems running the same DBMS (POREL [NEUH77]), different DBMSs on identical computers, etc.

In practice, a heterogeneous system will be useful if it proposes the following facilities :

- to allow the users to manipulate a database without having to worry about the distribution of data and the diversity of local systems,
- to provide the facility to manipulate the distributed database with different languages. So, each language can be more adapted for each usage,
- to make possible the integration of an existing database in a DDBMS without any reorganization of this new local database and without any modification in applications (keeping data and languages).

In order to build a system which corresponds to all these criteria, heterogeneity have to be managed for each componant level.

2. The SIRIUS answer to heterogeneity in DDBMSs

The first aim of the SIRIUS pilot project has been to develop a prototype of a homogeneous DDBMS called SIRIUS-DELTA [LEBI81]. From this prototype, we have experimented the heterogeneity; that for, we have connected heterogeneous computers and DBMSs through a local network in order to realize a prototype of heterogeneous DDBMS.

2.1. Survey of the heterogeneous prototype

SIRIUS-DELTA has been built with three minicomputers connected, in a first step, through point to point links. Each of these computers has the functions of data base management and distributed data base interrogation. A large computer supporting the MRDS relational DBMS and a micro-computer supporting the PHLOX network DBMS ([PHLO79],[PHLO80]) are connected to these three computers through the DANUBE local network developed at INRIA by the KAYAK pilot project. In the heterogeneous DDBMS prototype, these two last computers only have the data management function (see figure 1).

2.2. The prototype architecture

The prototype architecture is considered with regard to :

- firstly data base,
- secondly functionalities.

2.2.1. The data base architecture

This architecture is shown through the schemata describing the DDB components.

It is mainly made up of two levels : the global level, corresponding to the whole DDB and the local level corresponding to the DBs which compose it.

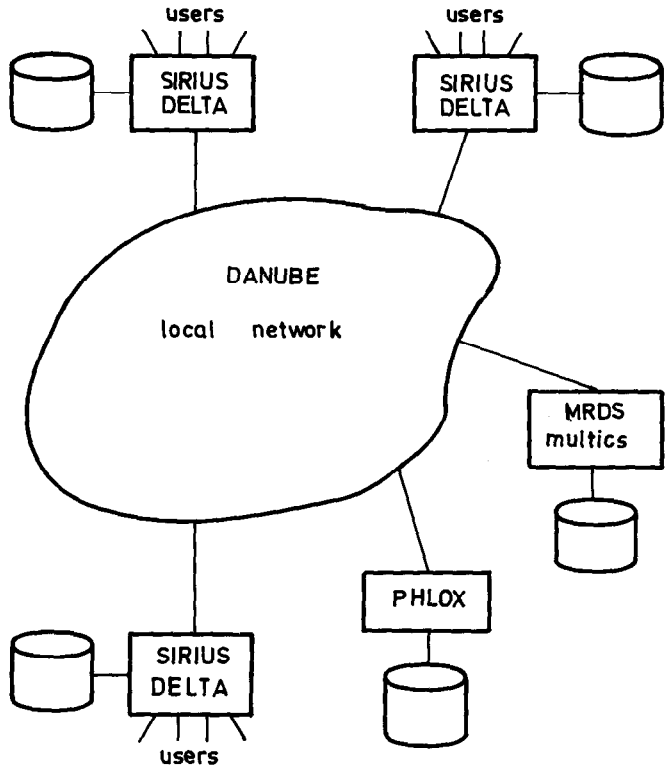


Fig. 1 : Heterogeneous DDBMS : SIRIUS-DELTA + PHLOX + MRDS

- The DDB global level :

  It is at this level that the DDB is considered as a whole, as a unique base, therefore, the DDB conceptual schema and all the associated external schemata (as for a classical data base) are included at this level [ANSI75], [TSIC77].

  The particularity due to the fact that the data base is distributed appears in the global internal schema. The global internal schema contains the description of the mappings between global and local data as well as the necessary localization and duplication rules. Global data are described as the result of a sequence of operations applied to local data. The global internal schema gives the way to rebuild global data from local data. It contains the description of sets of local data belonging to each site, that compose the DDB (the example in appendix shows the content of a GIS).

- The DDB local level :

  This level contains the main components of the DDB : the data bases called local DBs. The data are stored in the local DBs and the global level takes the part of a user for each of them. Therefore, in the DDB context, the local level includes an external schema, a conceptual schema and an internal schema per local base (see figure 2).

Local data are expected to contribute to the DDB, but not all local data do, and from the DDB point of view, local data may look heterogeneous.

The local external schemata are provided so that heterogeneous local data are presented and handled as homogeneous at the DDB level.

A local external schema allows the mapping between the two descriptions of local data contained respectively in the global internal schema and in a local conceptual schema.
If a local base is also used directly (in another way than via the DDBMS) corresponding external schema or schemata must be added (e.g. LES_{11} in figure 2).

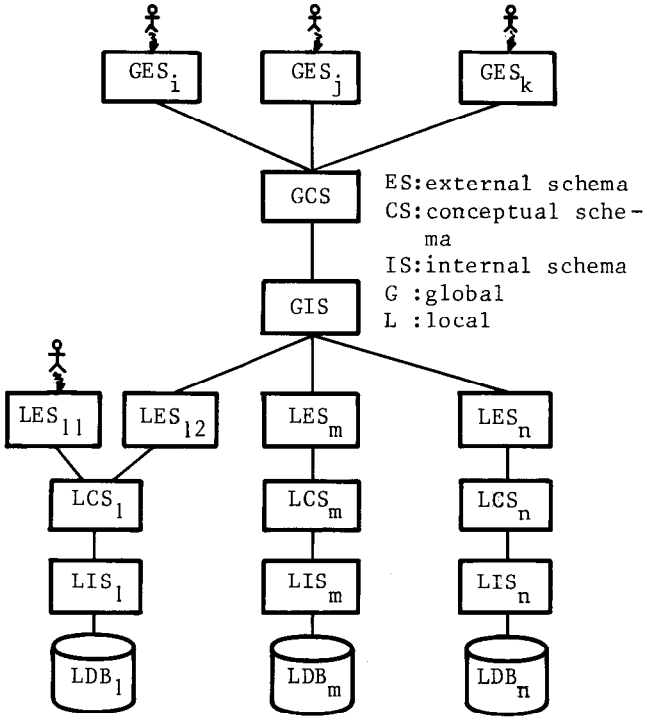


Fig. 2 : DDB architecture according to the schemata.

2.2.2. Functional architecture

In the heterogeneous prototype, four layers are constructed on top of a transport service as defined by the ISO reference model of open system architecture [ISO79].

The layers exist in the global and the local levels.

- DBMS layer : classical functions of data base management (query analysis at the global level, access to local data at the local level).
- SILOE layer : functions of data distribution management (query decomposition according to data location at the global level and interface with the local DBMSs).
- SCORE layer : function of distributed control (data base consistency and reliability).
- SER layer : function of distributed execution (remote process activation, data transfer between processes).

2.3. Definition of the pivot system

In order to co-operate with the heterogeneous DDBMS, each system must have a minimal set of functions.

This minimal set of functions is called pivot system. It is a virtual system as close a possible to the main existing systems.

The use of a common reference, the pivot system, has two advantages : it makes possible to realize a unique DDBMS at the global level -in expressing all the operations according to the pivot system- and to limit the number of translators as shown by the following figure (figure 3).

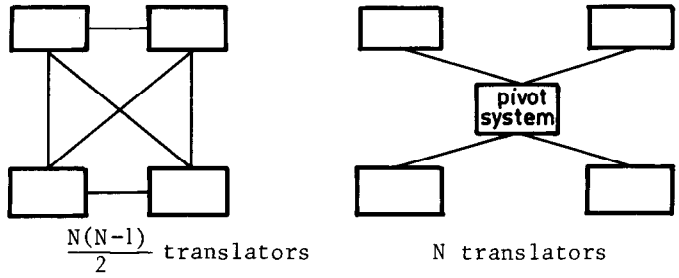


Fig. 3 : Advantage of the pivot system

The definition of the pivot system has a large impact over the system in its whole. If it is much more powerful than most of the existing systems, the missing functions will have to be developed on each system in order that it reachs the "pivot level".

If it is not much powerful, the expression possibilities of the global level should be loosed. This can lead to reduce the performances of local systems or the general functionalities. For example, a pivot system with only navigational manipulations would imply a very large number of messages, unacceptable over a network.

In our prototype, the pivot system consists of the set of the SER, SCORE and SILOE functions, as they have been defined in the SIRIUS-DELTA prototype of homogeneous DDBMS [LEBI81]. To integrate a system in this prototype, the local and/or global functions of SER, SCORE and SILOE have to be developed over this system, according to the part played by the system in the DDBMS. The global functions should be implemented if the DDB can be accessed from this system, the local ones, if this system manages data belonging to the DDB.

This integration should use, as much as it is possible, the existing services of the system. The maximal use of the abilities of the existing systems is one of our goals.

Here is an example concerning the transaction commitment : our pivot system includes the roll-back function. In fact, this function can be assumed in many ways : before look saving and write straight on the data base, differential files, or other method. The main point is that a local system, co-operating with the heterogeneous prototype, must satisfy the functional constraints defined in the pivot system.

If systems of different types take part in the DDBMS, then the DDBMS makes heterogeneous DBMS co-operate, that is to say DBMSs with different models and different manipulation languages.

A common model and a common language are defined for the prototype. They are called respectively pivot model and pivot language.

2.4. The pivot model

As we have pointed it out in section 2.2.1, many schemata exist together in the DDBMS. The global conceptual schema and the local conceptual schemata can use different models : hierarchical, network [CODA71], entity relationship [CHEN77], relational [CODD70]...

All these schemata have to be able to rely on a unique description of the DDB. This common description of the DDB will also be used as a reference by the pivot language.

We chose the relational model as pivot model because it has been proved in [DEMO77], that this model is compatible with any other one. As it describes the database in a unique and common way, the pivot model makes it possible to solve the problems of heterogeneous terminologies and heterogeneous data structures. All the data circulating on the network, have to be represented according to the informations kept in the pivot model.

The local external schemata make it possible to do the connection between the database description according to the pivot model and the database description according to the set of conceptual schemata (each of these conceptual schemata describes the objects of a local database that belong to the DDB).

The local external schemata contain informations about data structures and formats needed by the global level. Using these schemata, the local level of the DDBMS is able to present data in a unique way : the way described by the global conceptual schema.

This mecanism looks like the one of MULTIBASE with the DAPLEX language and the Functional Data Model [SHIP81].

2.5. - The pivot language : PETAL

Several languages exist in the DDBMS :

- the languages used by the DBMSs from which the DDB can be interrogated ; they are called global external languages,
- the data manipulation languages of the DBMSs that manage the local databases ; they are called local external languages.

A user of the DDB submits a query with the global external language he uses currently. This query is analyzed, translated into an internal language and decomposed into sub-queries. The sub-queries are sent to the local DBMSs that manage the data concerned with the sub-query. Therefore, local DBMSs have to be able to understand the sub-queries they receive from the global level. A pivot language provides a language that every local DBMS can understand and it is a solution that minimizes the number of translators to be implemented. Each sub-query just have to be translated into the pivot language at the global level. At the local level, the sub-query will be translated into the local external language existing on this site (see figure 4).

The pivot language relies on the pivot model that keeps the description of data manipulated by the sub-requests in pivot language.

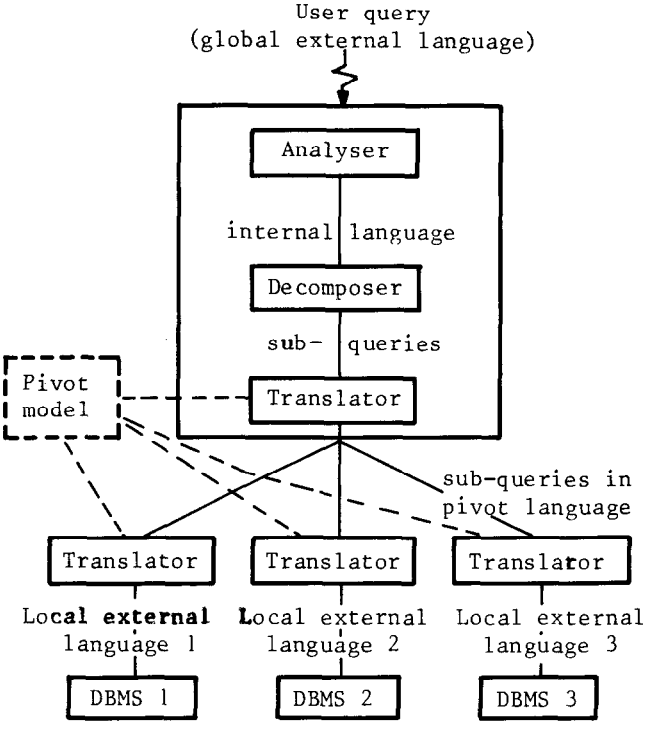


Fig. 4 : Pivot language

The example, presented in appendix, shows the transformations of queries and sub-queries which appears in figure 4.

2.5.1. Choice of a pivot language

In a distributed environment , working on isolated elements of the data base would lead to many messages between the sites (this is of very high cost). Therefore it's better to work on sets of data. Consequently, we have chosen a relational pivot language [GLOR81].

Two types of relational languages are to be considered :

- algebraic languages : these languages are quite procedural andthey state series of "elementary" operations (join, project, etc) ;
- relational calculus languages : they approximate the predicate calculus (like QUEL). They provide a dense and synthetic expression of the request.

A choice has to be made between pivot languages based respectively on relational calculus and on algebra. This choice depends on the external languages of the DBMSs connected on the network.

We first have to notice that a query, expressed with a predicative language, is decomposed into a series of operations on sets and on relations. Consequently, the interface between a low-level pivot language (algebraic) and a predicative language of a local DBMS, will be rather simple to realize.

Nevertheless, in this case, some tools of a local DBMS supporting a relational calculus language will not be used, because it is difficult to regroup the sub-queries in an optimal way, since they have been very much decomposed.

On the contrary, if the pivot language is synthetic, we shall have to add to local DBMSs supporting elementary operation languages, an interface that will decompose the sub-queries in pivot language into a set of elementary operations that the local DBMSs can understand.

The PHLOX DBMS uses a navigational language and has a relational interface, implemented through a minimal set of relational operators ; the SIRIUS-DELTA homogeneous prototype uses a language which is easely decomposed into a series of operations on sets and relations ; so we chose an algebraic pivot language called PETAL.

2.5.2. PETAL

PETAL is composed of the three relational operators : select, project, join, and the three operations on sets : union, intersection and difference, for the consultation. For the update, it is composed of three operators : modification, creation and suppression. This language uses a tree structure in order to describe the chaining of operations. Each tree, corresponding to a sub-queries is described using the post fixed notation.

3. PHLOX

The aim of the PHLOX project is the design and the realization of DBMSs for micro-computers. Two prototypes are running on INTEL 80/86 :

- PHLOX1 for individual micro-computer (mono-user, mono-server) and
- PHLOX2 for a dedicated server in a network (multi-user, mono-server).

A third prototype, the PHLOX3 distributed DBMS (multi-user, multi-server) is being studied.

3.1. The PHLOX system's architecture

The system is multi-levelled ; each new prototype is being built adding new levels to the previous one.

We shall take an interest in PHLOX2 since it is this system that can be used as a server in the heterogeneous SIRIUS-DELTA prototype.

PHLOX relies on a dedicated operating system which consists of 3 levels :

- a sequential and direct access file management system,
- a B-tree management,
- a virtual context management which deals with consistency and resiliency.

In administration,PHLOX offers the possibility of describing a new data base schema using the DBTG's network model [CODA71]. In manipulation, a navigational data manipulation language is available.

3.2. Connection of PHLOX with the heterogeneous DDBMS SIRIUS-DELTA

The functionalities of PHLOX are very different from the pivot structures of the protocol :

- the internal coding of data is different,
- the access schema is a network schema,
- the navigational primitives allow only step by step manipulations of a database.

In order that it can cooperate, as a server, with the DDBMS, PHLOX has to provide the following facilities :

- to convert the exchanged data into ASCII code,
- to take into account a relational schema, which is a subset of the global schema of the distributed database,
- to perform the relational operators as they are expressed with the pivot language PETAL,
- to take part in the executive and concurrency distributed protocols,
- to communicate with the other sites through the DANUBE network.

Converting data and communicating are classical problems for which solutions are well known. The concurrency systems of SIRIUS-DELTA and of PHLOX rely on the same principles [LELA78], and though they are implemented in different ways, they can cooperate rather easely.

3.3. Relational interface

The most critical problems PHLOX has to solve is translating PETAL relational queries into PHLOX navigational queries. This translation requires, besides, a mapping from the relational schema of the local database, as it is viewed by the global DBMS , to the internal network schema used by PHLOX for accessing data.

The solution that have been chosen consists in adding to PHLOX a relational interface in the form of new levels. The advantage of this implementation is that the relational operators can be used when PHLOX runs alone (a relational DML analyser has been written in order to make it possible to use the relational tools). The mapping from a relational to a network schema is done at the creation of the description of a database schema.

3.3.1. Mapping : relational-network

The database administrator can describe a new database according to the relational model. He defines : the names of the relations, the names of the attributes and their characteristics. Then the system processes the mapping of this schema into a network schema.

3.3.2. Execution of PETAL queries

PETAL queries are processed through two levels, added to PHLOX (see figure 5) :

- a set of relational operators,
- a PETAL interpretor.

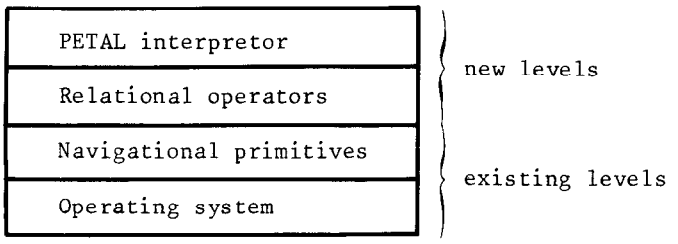


Fig. 5 : New levels in PHLOX

a) Relational operators

This module includes the three relational operators : project, select and join ; it is necessary to associate with it the operations on sets : union, intersection and complement.

Inserting, suppressing and updating is made easely thanks to the existing primitives : insert, suppress, link, unlink, modify. The execution of the relational operators uses the tools of the system, mainly the B-tree management and the navigational data manipulation primitives. The physical access to data by PHLOX makes possible to minimize the reading and writing in many cases :

- temporary relations which result from a selection are represented by a B-tree and tuples can be retrieved into the original relation though the selected tuples are not rewritten.
- indexes defined on some data types make it possible, in certain cases of selection, not to read the whole data or not to read any data at all.
- some joins can be done using the father-son access of the network schema, only the records answering the question are read.

b) PETAL interpretor

From a query expressed with the PETAL pivot language, the interpretor manages the execution and the chaining of the relational operators. It also realizes the mapping from the external names of relations and attributes corresponding to the pivot model into the local names corresponding to the conceptual local schema described by the tables of PHLOX.

We plan to include optimization functions in this interpretor. Their part is to reorganize the tree of the request, taking into account the physical method of access to data, in order to minimize the reading and writing.

4. Solution of heterogeneity by SIRIUS-DELTA and the system supporting MRDS

4.1. SIRIUS-DELTA implementation

4.1.1. At the global level

The DDB can be accessed from a SIRIUS-DELTA site. As we have built the pivot system from the functions yet experimented in the homogeneous prototype, the three systems co-operating in SIRIUS-DELTA should be able to take into account the heterogeneity problem with only slight modifications. Nearly all the global functions of the pivot system yet existed. Only the interfaces with the PETAL pivot language and the pivot model were yet to be realized. The global external language existing in SIRIUS-DELTA is FRANCAIS. The existing SILOE layer analyzes the FRANCAIS query and translates it into a chaining of operations on sets and relations. It decomposes this query and produces sub-queries, expressed with the same operators. Therefore, the translation into PETAL is very easy as PETAL and the analyzed query both use the same operators. We had also to implement the mapping between the hierarchical model used by SIRIUS-DELTA and the relational model (the pivot model).

The translation of the relations and attributes names as they are known in SIRIUS-DELTA into the common terminology used by the pivot system is supported by the pivot model.

4.1.2. At the local level

Each of the three systems of SIRIUS-DELTA manages data belonging to the DDB. Nearly all the local functions already existed. In the local SILOE, the translation function of PETAL into FRANCAIS (the local external language supported by the local systems of SIRIUS-DELTA) was to be added.

FRANCAIS is a predicative language. Therefore, we have to try to regroup in a maximal way the PETAL operators in order to formulate a FRANCAIS request and use, if it is possible, all the capacities of the local system. A translation and optimization function was implemented that for.

The translation of the common terminology (the pivot model one) into the local system terminology is supported by a mechanism of local external view. Informations, concerning the structure wanted for the data taken out of the base, are also stored in that local external view.

Thus, data circulating on the network are structured in an unique way.

The heterogeneity of data presentation is solved in that manner. Before sending data by the network, the local SILOE layer converts them into the common structure. With each data stream, local SILOE sends informations concerning the data structure and names. These data can be manipulated by the receiver site as if it was its own data.

4.2. The local level supported by MRDS

In our heterogeneous prototype, MRDS plays a part only in the local level. That is to say that it manages data belonging to the DDB. The local functions of the pivot system : local SER, local SCORE and local SILOE are to be implemented. These function were realized by using existing services on this system.

MRDS supports a QUEL-like language : LINUS. The translation of PETAL into LINUS was rather simple. As MRDS supports a relational model, the mapping between the MRDS model and the pivot model was very simple.

Only the data presentation was different. MRDS uses fixed length data whereas the common structure is constructed with variable length data. By sending between systems, over the network, informations concerning the data structure (and in particular the maximal data length), this problem has been resolved.

Conclusion

The SIRIUS-DELTA experiment about heterogeneity shows the usefulness of a pivot system when systems of different type have to cooperate. The design of such a common standard makes possible to reduce the number of translators and interfaces which have to be implemented on each system running in the DDBMS. Our aim has been to design a pivot system so that each existing system can be easely adapted to it, and we have tried to use the existing functions of the different systems the most we could. The realization of the heterogeneous prototype proves that such a system is feasible. This experimentation is more especially probative as the interconneted systems are quite different : they are running respectively on very large, mini and microcomputers ; they rely on three kinds of DBMSs : relational, hierarchical and network.

Appendix : An example of query processing in the heterogeneous DDBMS SIRIUS-DELTA

An example of DDB and of query submitted to the heterogeneous DDBMS will help to understand the content of the different global schemata and the translations of a query.

A1. Global schemata of a DDB

A1.1. Global conceptual schema

The GCS in our example consists of two relations:

- TOWN(NT,NAME,LOCATION,ALTITUDE), NT is the key.
- HOTEL(NH,HOTEL-NAME,PRICE,NB-ROOMS,NT), NH is the key.

A1.2. Global internal schema

The GIS describes the distribution of data. Our database is composed of three local data bases : LDB1, LDB2 and LDB3. The GIS contains the description of the plots of relations and the localization of each plot :

- Relation TOWN consist of three plots :
  T1(NT,NAME,LOCATION,ALTITUDE)
  T2(NT,NAME)
  T3(NT,LOCATION,ALTITUDE).

  TOWN=T1 when ALTITUDE > 800
  TOWN=JOIN(T2,T3) through NT else.

  T1 belongs to LDB1,
  T2 belongs to LDB2,
  T3 belongs to LDB3.

  The decomposition and distribution of the relation TWON is shown in figure A1.

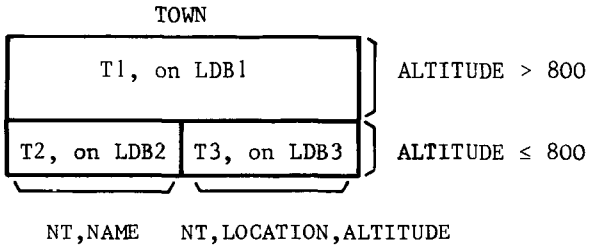


Fig. A1 : Relation TOWN

- Relation HOTEL consists of two plots :

H1(NH,HOTEL-NAME,PRICE)
H2(NH,NB-ROOMS,NT)

HOTEL=JOIN(H1,H2) through NH

H1 is on LDB1,
H2 is on LDB2.

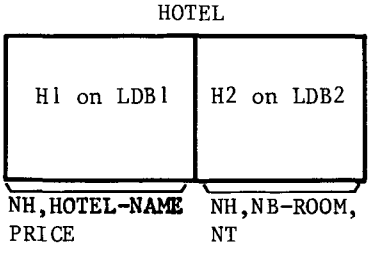


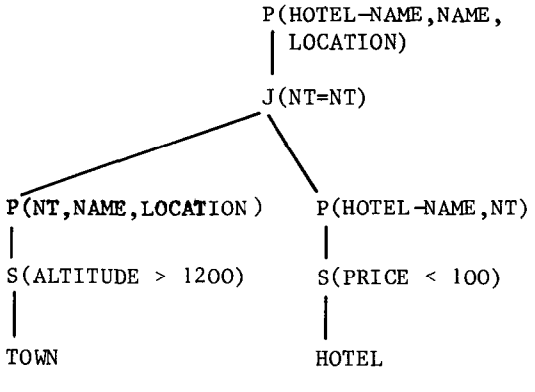
Fig. A2 : Relation HOTEL

A2. An example of query

We will analyse the following query : retrieve the hotel names, town names and town locations for the hotels the price of which is lower than 100 and which are located in a town with altitude higher than 1200.

A2.1. On the global site

The query is first translated into the tree of figure A3.



P = project
S = select
J = join

Fig. A3 : Tree of the global query

Then the relations TOWN and HOTEL are replaced by their decomposition and the tree is optimized (suppression of useless plots, etc.) according to the GIS (see figure A4).

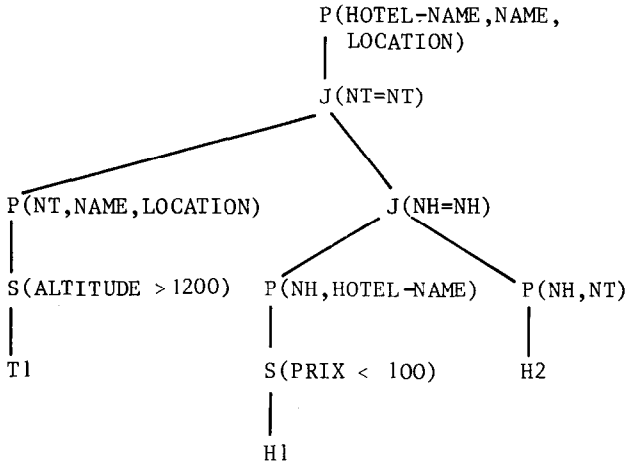


Fig. A4 : Intermediate tree of the query

This global tree is decomposed into local actions. A local action is composed of :

- a sub-tree wich corresponds to a sub-query,
- conditions on this sub-query (site where to perform it, etc).

The left branch of figure A4 will become the tree of a local action (figure A5) this local action will have to be performed on the site of LDB1.

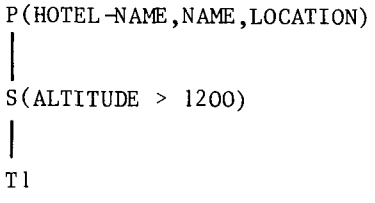
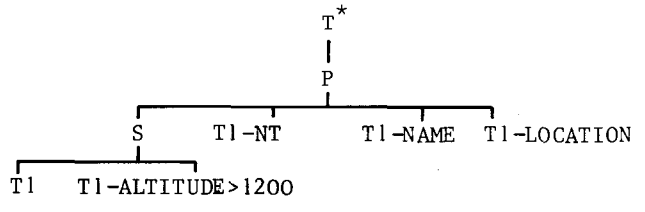


Fig. A5 : Tree of a local action

Each local action is translated into a PETAL query. For example, the local action of figure A5 will be translated into the following tree (figure A6) :



*T = transfer : the result of the sub-request has to be transfered to an other site.

Fig. A6 : PETAL sub-tree

A2.2. On the local sites

A local site of the DDBMS receives PETAL sub-trees and translates them into the local DML.

The sub-query in figure A6 can be translated by PHLOX, SIRIUS-DELTA and MRDS at the local level:

- PHLOX changes the names of relations and attributes in the tree, according to the external local view which corresponds to a plot of the global conceptual schema. The new tree is then performed by the PETAL interpretor of PHLOX.
- SIRIUS-DELTA translates the PETAL sub-query into a "Français" query according to the external local view.The sub-query of figure A5 will become a french sentence, the english translation of which is : List town with alt > "1200" number town-name location.
- On a MRDS site, this sub-query will become in LINUS :
  Range (T Town)
  Select T.nb T. name T.area where
  T. height > 1200.

References

[ANSI75] ANSI/X3/SPARC study group on data base management system : Interim Report. FDT, Bulletin of ACM SIGMOD, 1975.

[BAER80] BAER, GARDARIN, GIRAULT, ROUCAIROL, The two step commitment protocol ; modeling, specification and proof methodology, Technical Report, IP University of PARIS VI, May 1980.

[CHEN77] CHEN P.P., The entity relationship model. A basis for the enterprise view of data, NCC conference, 1977.

[CODA71] DATA BASE TASK GROUP OF THE CO.DA.SY.L. April 71 Report ACM, New York, 1971

[CODD70] CODD E.F., A relational model of data for large shared data banks, CACM, June 1970.

[DEMO77] DEMOLOMBE R., LEMAITRE M., Rôle d'un modèle commun dans la conception et l'utilisation d'un SGBD réparti : Analyse des principaux modèles, Journées AFCET, Base de données réparties, Paris, Mars 1977.

[GLOR81] GLORIEUX A.M., STANGRET C., L'hétérogénéité dans le SGBDR SIRIUS-DELTA. CIL'81, Barcelone, Juin 1981.

[ISO79] International Organisation for Standardization, ISO/TC9/SC16 : Open System Interconnection.

[LEBI81] LE BIHAN J. et al., SIRIUS DELTA Distributed Database System, 5th Berkeley Workshop on Distributed Data Management and Computer Networks, 1981.

[LELA78] LE LANN G., Algorithms for distributed data-sharing systems which use tickets, Proc. 3rd Berkeley Workshop on Distributed Data Management and Computer Networks, 1978.

[NEUH77] NEUHOLD E.J., BILLER H., POREL : A Distributed Data Base on an Inhomogeneous Computer Network. Proc. 3rd Intern. Conf. on Very Large Data Bases, TOKYO, October 1977.

[PHLO79] DEL VECCHIO B., FAIDHERBE J., PENNY P., Definition et réalisation d'un noyau de SGBD pour micro-ordinateur.Mémorandum at l'Institut d'Informatique d'Entreprise, CNAM, PARIS, 1979.

[PHLO80] FERRIER A., KAEPPELIN F., Définition et réalisation d'un noyau de sécurité pour un SGBD implanté sur un micro-ordinateur dans le cadre d'un réseau. Memorandum at l'Institut d'Informatique d'Entreprise, CNAM, PARIS, 1980.

[ROSE78] ROSENKRANTZ D., STEARNS R., LEWIS P., System level concurrency control for distributed database systems. ACM TODS, vol. 3, N° 2, June 1978.

[SHIP81] SHIPMAN D.W., The Functional Data Model and the Data Language DAPLEX, ACM TODS, vol. 6, N° 1, March 1981.

[TSIC77] TSICHRITZIS D., KUIG A., The ANSI/SPARC DBMS, framework-report of the sudy group on data base management systems. Technical note 12, CSRG, University of Toronto, ONTARIO (Canada), July 1977