Dr. Eng. V. PESCARU
I. CATONA
C. DRAGOMIR
V. DUMITRACHE
M. UNGHIANU
P. USVAT
V. VICOLEANU

Centre of Informatics and
Cybernetics for Planning

Development of Data Bank Management Systems with Facilities for Calculations on Tree-Like Information Structures

In this study, starting from the emphasis laid on the main shortcomings of the management systems for generalized data bank, the authors maintain as a solution to eliminate these deficiencies (eventually an intermediate solution untill the achievement of new data bank management systems) the prolongation of the facilities of generalized data bank management systems with facilities oriented towards groups of users, by achieving standard functions, by restricting to some extent the sphere of generality but eliminating the deficiencies presented.

This opinion is maintained by the achievement, as part of the Centre of Informatics and Cybernetics for Planning of a management software for relational data bases called DBMS SOGER (SOCRATES developed with facilities for relational data base management based on the generalized management system for SOCRATES data bank, supplied by C.I.I.-France.

DBMS SOGER is characterised by noticeable results obtained in its exploitation within the planning data bank.

The utilization of data banks as highly developed informatic systems able to meet, to an ever greater degree, the users' present and future requirements, represents one of the main trends in improving the computer-based information systems.

The stage that informatics has covered since 1965 is strongly characterized by the concern to introduce and use data banks, to continuously improve their management systems (DBMS). One may say that, during this period,

a considerable qualitative leap has been made in the methods and techniques of data organization and structuring capable of turning to account the special facilities provided by up-to-date computers.

The efforts made the world over at expanding the use of informatic systems have also been materialized in building generalized data bank management systems meant to take over some of the programming tasks, necessary to ensure an integrated data management.

Although over the past years there has been perceived an intensification in the preoccupations to develop the data bank management systems software, the qualitative results have not yet come up to the users' complex requirements level *) [2], [4], [10].

As a rule, the share of standard tools within the DBMSs is still small in comparison to the share of programs which ought to be worked out in order to suit the concrete demands of the data bank users.

However, one should mention the quantitative leap noticed in the last few years demonstrated by the fact that the number of informatic systems based on implemented data banks has increased from several hundreds to several thousands. At the same time, in this wide process of data bank implementation the main difficulties have come out with a view to satisfying the users' concrete demands, as well as the need to improve data bank management systems.

As a consequence of the preoccupations existing throughout the world for qualitatively improving data bank management systems, it is to be expected in the coming 4—5 years [2], [6] that there will be built systems which will eliminate the main deficiencies that have resulted so far.

In the present paper, starting from the emphasis laid on the main shortcomings of generalized data bank management systems, the authors advance as a solution to eliminate these deficiencies (a possible intermediate solution until the construction of new DBMSs) the extension of the facilities provided by generalized data bank management systems with groups of user — oriented facilities, by achieving standard functions, by restricting, to a certain extent, the degree of generality but doing away with the above mentioned shortcomings.

This view is supported by the practical achievement within the Informatics and Cybernetics Centre for Planning in Bucharest of a management software for relational data bases called DBMS SOGER (SOCRATES developed with facilities for a relational data base management) based on the SOCRATES generalized data bank management system.

DBMS SOGER is characterized by considerable results obtained during its exploitation within the data bank of planning belonging to the State Planning Committee — Romania.

1. MAIN SHORTCOMINGS OF THE EXISTING GENERALIZED DBMSs

From among the shortcomings of the existing DBMSs, we would like to point out some, which we consider to be worthy of note, also underlined in [1], [2], [4], [6]:

*) the term "user" stands for the end user, the non-informatic user.

a) *Special effort for ensuring an efficient implementation.*

The reasons for which the implementation effort is considered a special one are the following:

— a thorough knowledge of the DBMS is necessary on the part of the data bank administrator (DBA).

— the design of the virtual structure of the data base/bases *) is decisive both for data bank performances in general, and for a subsequent effort to redesign it; this design directly depends on the DBA's knowledge and experience, which are not easily to be acquired at all.

— DBA must write by using the enquiry language of DBMS a number of programs, constituent parts of the utilization function: programs for introducing information into the data bank, programs for interface with programmers in high-level languages, programs for allowing the use of conversation by non-programmers etc.

These programs must be highly performative, as the succes of data bank implementation extensively depends on them, but in order to accomplish the necessary performances the DBA must have a thorough knowledge both as regards the management system proper, and the operating system one works with.

In keeping with DBA's knowledge and experience these programs are, to a smaller or greater extent, dependent on the structure of the virtual level of the data base description; the working out of programs as little dependent as possible on the virtual structure leads to a great economy of programming effort both in the experimental stage when changes of virtual structures often occur and in the reorganizations required by the exploitation stage.

Once again DBA's experience has the last say.

— Programs of applications written in the enquiry language of the DBMS used contain a high degree of redundancy; it necessarily follows that based on the knowledge of the needs of application programs, DBA should put at disposal a series of primitive functions which will lead both to the elimination of the respective redundancy and to the increase in program performances;

— Under certain circumstances in order to achieve ever better performances, DBA has to combine the enquiry language facilities with programming in the assemblage language of that exploiting system;

b) *Users do not have at their disposal a language of formulating demands which is easy, manageable and independent of the data structure.*

With most of the existing systems the logical model the user must be familiar with is very complex.

According to those presented in the technical literature [1], [4], [5], the hopes of eliminating this shortcoming converge towards using a relationa logical model for data bases and placing a relational calculus-based language at users disposal. Actually, most of the research efforts in the field of data banks, are made in this direction [2].

---

*) there are used the base and bank definitions DBMS given at [9].

c) Generalized management systems for data banks existing at present [6] *mainly provide facilities only for obtaining the data stored in the data bank.*

There is no possibility of setting out some rules for the retrieval of information and the automatic maintenance of information coherence; [6] indicates us an example in support of this idea the necessity that by modifying the unitary price of an item, the up dating of all data items dependent on it should be automatically carried out.

With such facilities, users would have at their disposal a DBMS capable of simulating and analysing the effect of decision-making.

Starting from the analysis of the SOCRATES system, [6] suggests as a solution the creation of a semantic language where such relationships can be written with adequate sequences for maintaining coherence called "methods".

2. A SOLUTION FOR DIMINISHING THE SHORTCOMINGS OF GENERALIZED DATA BANK MANAGEMENT SYSTEMS

From the deficiencies mentioned at point 1., there follows the neces sity of building DBMSs which:

a) should permit an implementation with a minimum contribution on the part of DBA;

b) have a non-procedural language regardless of the way data are structured and easily to be manipulated by users;

c) permit data modification by automatically maintaining the data bank coherence, therefore they should also be a simulation instrument.

The creation of such DBMSs with a high degree of generality is difficult and as results from the technical literature [2][6], [12], further essential improvement of DBMSs in the way shown above, is to be expected in the next 4—5 years.

In these circumstances, we consider that one of the solutions to be taken into account is that based on a generalized DBMS, completed (developed) with standard program modules which ought to comply with the fore-mentioned requirements, but with a certain restriction on the sphere of generality oriented towards the needs of certain categories of users, of applications respectively (management software for user-oriented data banks).

For example, specialists engaged in the development and utilization of certain DBMSs could also produce a programs package (a DBMS standard module) oriented towards some enterprise's management problems by using the conversational mode as well.

In this way DBA's tasks would not consist in designing the virtual structure, in writing programs, data base loading, and so on., but in setting some parameter values and adequately using some tools achieved at a top quality level. In this case, the designing and programming effort respectively, could be reduced by 50—60%, mainly limiting itself to adaptations to concrete conditions.

Naturally, users have at their disposal all the facilities of the system proper, but they only use them for completing certain specific aspects, the

chances of success in implementing the DBMS and generalizing it at the level of several enterprises being therefore much increased.

On the basis of these considerations, and having as a main goal the building of data banks for planning, as it has been previously stated, within the Informatics and Cybernetics Centre for Planning such a user-oriented software has been created, which, to some extent, meets the three demands presented at the beginning of this paragraph.

In particular, this software is used for the data bank management of planning, but it can be used by a larger category of users, especially in the tivities bearing an economic character (planning, statistics, finances, foreign and home trade etc.) carried out at various levels of national economy organizational structure.

From the users' logical point of view this softwaremanages a relational-data base for the reason of which it is called "SOGER".

3. THE CONSTRUCTION CONCEPTION OF THE SOGER SYSTEM

The SOGER system is conceived to suit the three demands of improving generalized data bank management systems, presented at [2] and marked with a, b, c,; we shall describe the way it meets these demands.

a) The system is conceived as a package of parametric applicative programs accomplished on the basis of a generalized virtual structure defined with the aid of the SOCRATES description language, which in its turn admits of certain parameters.

Hence it follows that experts in informatics who wish to implement the system in a domain where their needs can be satisfied have but to establish based on the analysis of applications the respective parameters, without having the least knowledge of the SOCRATES system.

Thus, it is met the necessity of diminishing, to an ever greater extent DBA's tasks and the dependence of the system's performances on DBA's, training and experience respectively.

b) To satisfy the second need, the SOGER system is designed in keeping with the relational data bases theory.

For users, the data base logical model is a relational model, therefore data base is a collection of relations for them. A relation is regarded by the user as a bidimensional table with lines (n-uplets) and columns (domains).

In this view the user has available for use a relational calculus-based language by means of which he operates in conversational or batch processing on the data base relations.

The user also has the possibility of defining new relations, new domains of a relation etc. This logical model put at the user's disposal, consisting in looking upon the data base as a set of bidimensional tables is very simple and the language he operates with on this logical model for interrogation, updating, calculi etc. may be entirely independent of the way data items are organized.

Thus, the user does not have to know the fact that behind this very simple logical model there hides a complex, network-type structure for data representation.

It results that from users' viewpoint nothing changes if new system versions are being created, if data virtual structure or access strategy get modified;

c) The SOGER system is also conceived as a simulation instrument for users, one of analysing the influences the taking of certain decisions would have.

By the system's defining function it is possible to define domains of certain relationships as resulting from arithmetic and/or logical operations on other domains.

Of course, a domain that results from calculation may also in its turn take part in the calculation of other domains etc.

Based on the way these domains are defined, the system automatically ensures the maintenance of data base coherence. Thus, in the case of some value updating in the data base, there are automatically transmitted the influences on all data items which are in certain calculus relations with the modified data.

Optionally, both in batch and conversational processings, the display of the route covered, of the values before and after the modification respectively, can be obtained.

Naturally, these influences are transmitted according to some priorities which automatically establish in order to ensure that the updating of a relation domain should be done before the updating of the domains dependent on it.

During the simulation process the user can obviously change the calculus relations as well, in a conversation included, by directly introducing the changes into the display.

The data bank possibility by which it represents a simulation instrument too, is mainly ensured by DBMS SOGER standard module which permits the effectuation of arithmetic and logical operations (standard module for calculi).

4. PRESENTATION OF THE SOGER SYSTEM MAIN FUNCTIONS

4.1. *The system's general functioning principles* are as follows:

a) the data base is considered to be a set $\mathfrak{R}$ of relationships, of various orders;

b) the elements of the set $\mathfrak{R}$ can be divided into three categories:

1) a relationship $\mathfrak{T} \in \mathfrak{R}$ in which all the characteristics of a text type (the classified lists) are coded.

The domains of this relationship are: the code of the list type, the code of the position in the list, the full denomination, the abbreviated denomination and general information.

The first two domains make up the relationship primary key $\mathfrak{K}$:

2) A set of relations for representing numerical values. The first domain of these relations is denoted by $\mathfrak{K}$ and stands for the primary key.

The other domains are domains that may contain secondary keys for connections with other relations., especially with the $\mathfrak{T}$ relation, exogenous

numerical values, or numerical values which can be deducted on certain calculus rules;

3) Binary relations having two domains marked (K_1, K_2) which have as values elements of the domain k from the previous relations. These binary relations define structures on the set n-uplets of the previous relations.

One or several binary relations can be associated with the $\mathfrak{R}$ relation.

One binary relation marked with $\mathfrak{s}$ is implicitely associated with one relation from the second category;

c) The set of relations defined in the data base is sequentially coded starting with value 1; references to relations are made by associate codes.

The domains of a relation get an identification number representing a secondary key of connection with the $\mathfrak{R}$ relation; references to domains are made through these codes;

d) The coding principle for lists is represented by the insignificant, serial coding within every list type. In order to ensure preferential orders for the display of lists binary structures are used.

e) All SOGER system functions can be used both in batch and in a conversation.

The working principle in a conversation is represented by the testing (entry) of the order wanted on the display and the starting of a dialogue in which the user is interrogated and, depending on necessities, he has some auxiliary data at his disposal in order to be able to respond (the list of the already defined relations, the domains of a relation, classified lists etc.). This kind of work enables any uninitiated user to communicate with the data bank and obtain the information he wants.

In batch processing, one can simulate work in the conversational mode, which is more inconvenient, or one may apply to the respective functions specific to this mode of operation.

f) Data of a domain are homogeneous, therefore a possible calculus rule is valid for all the domain values.

g) Relationships, the way they have been defined, correspond to CODD's third normalized form, as presented at [2].

4.2. *SOGER system structure*

The main components of DBMS SOGER are the subsystems: for meta data base management (MDM), for data management in the data base (DM) and for data base reorganization (DBR).

The general scheme of functioning and using of DBMS SOGER main components is shown in Fig. 1.

4.2.1. **Meta data base management subsystem (MDM)**

According to the definitions at [8], by meta data we mean the information describing the data of the data base as well as the interdependence between them. In order to define the same thing in [3] it is used the notation DD/D (Data Dictionary/ Directory) while other reference materials use the notions of Data Catalogue or Library of Meta Data.

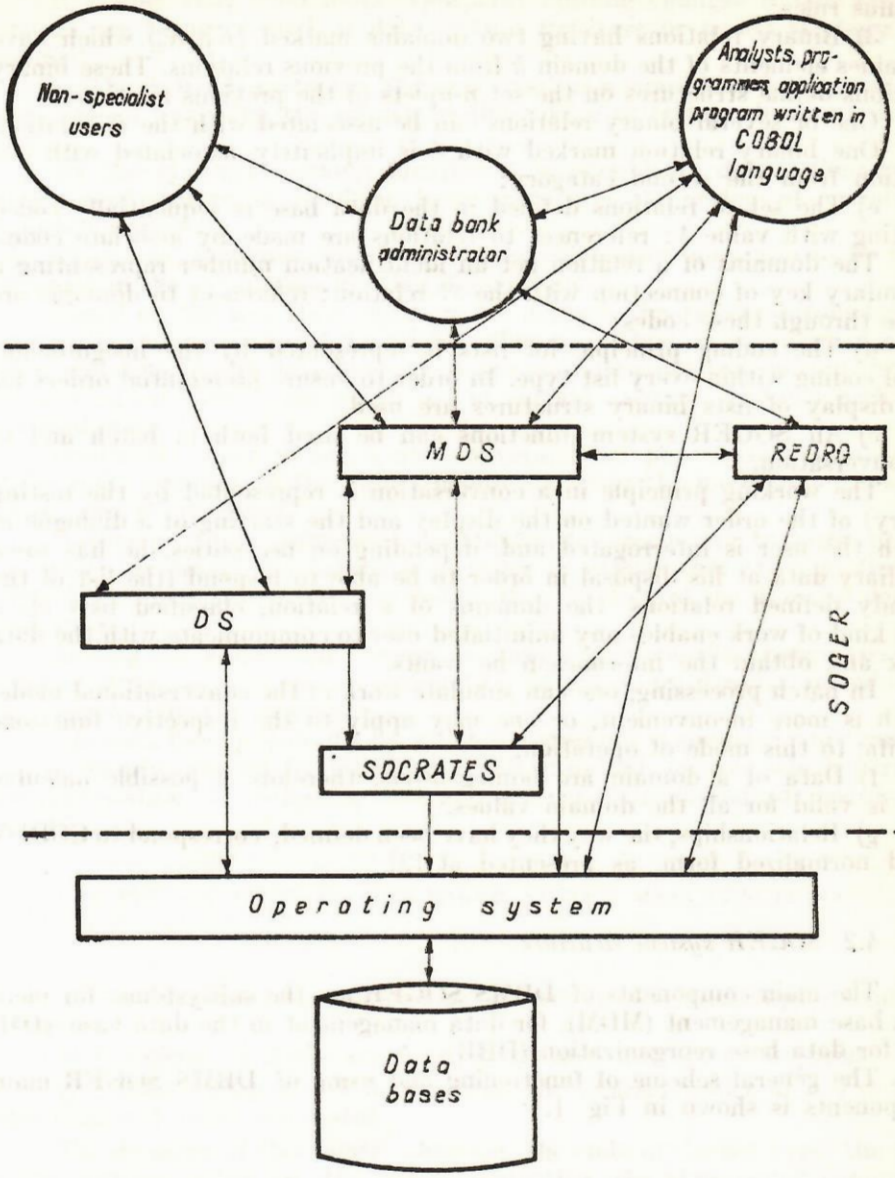

Fig. 1

Although the notations are different, there is a general consensus as regards this subsystem's contents and role. According to this general consensus, MDM subsystem has two main functions:

— manages information concerning the data bank contents necessary to users (the list of the categories of information existing in DB, the list of output records which can be automatically obtained and so on.);

— manages information concerning DB contents necessary to application programmers, to analysts as well as to the data bank administrator.

Out of the information addressed to specialists we could mention:

1) the data base virtual description;

2) description of the structures of the logical records required by application programs and of the way they obtain from the virtual structure;

3) description of the relations existing in DB (the elements of set $\mathcal{R}$);

4) association of a binary relation with another one, which brings about a structure on the set of n-uplets pertaining to the respective relation and, at the same time, association of certain significances with the structure levels;

5) definition of some attributes associated with every domain of a relation, such as:

— the type of domain (primary key, connection key with other relations, domain with exogeneous numerical values, domain that has resulted from arithmetic calculi made on other domains, domains containing aggregable values, domains containing values which are to be deducted on the basis of associated shares etc.);

— the code of the unit of measurement attached to numerical values;

— the number of decimal fractions which signify the value, the number of decimal fractions used in calculations and the number of decimal fractions used in standard reports;

6) Definitions of the calculus formulae, of the arithmetic rules respectively, for obtaining a domain declared calculable. Those rules apply to all the domain values;

7) Description of standard reports which can be automatically supplied, descriptions meant for programs that obtain the respective records;

8) Description of the significance of values belonging to the standard logical structure used for the entry of data to the data base.

The MDM subsystem is made up from a series of programs which make possible the management of the descriptions briefly discussed earlier. The subsystem permits a dynamic management of the data base relations, the addition of new domains to an already existing relation, the deletion of certain domains, the addition or deletion of certain relationships, the updating of calculus formulae etc.

The view and design of this subsystem is one of the key elements in building the SOGER system.

Indeed, the existence of this module which achieves the management of a set of descriptive parameters allows the writing of some programs that should function on the basis of their interpretation, which should be, therefore general and not dependable on data bank dynamics. Most of the programs, component parts of SOGER system are worked out according to this principle.

4.2.2. Subsystem for Data Management in the data base (DM)

This subsystem consists of generalized programs based on using the descriptions managed by MDM system. These programs ensure the next functions: (also presented in Fig. 2):

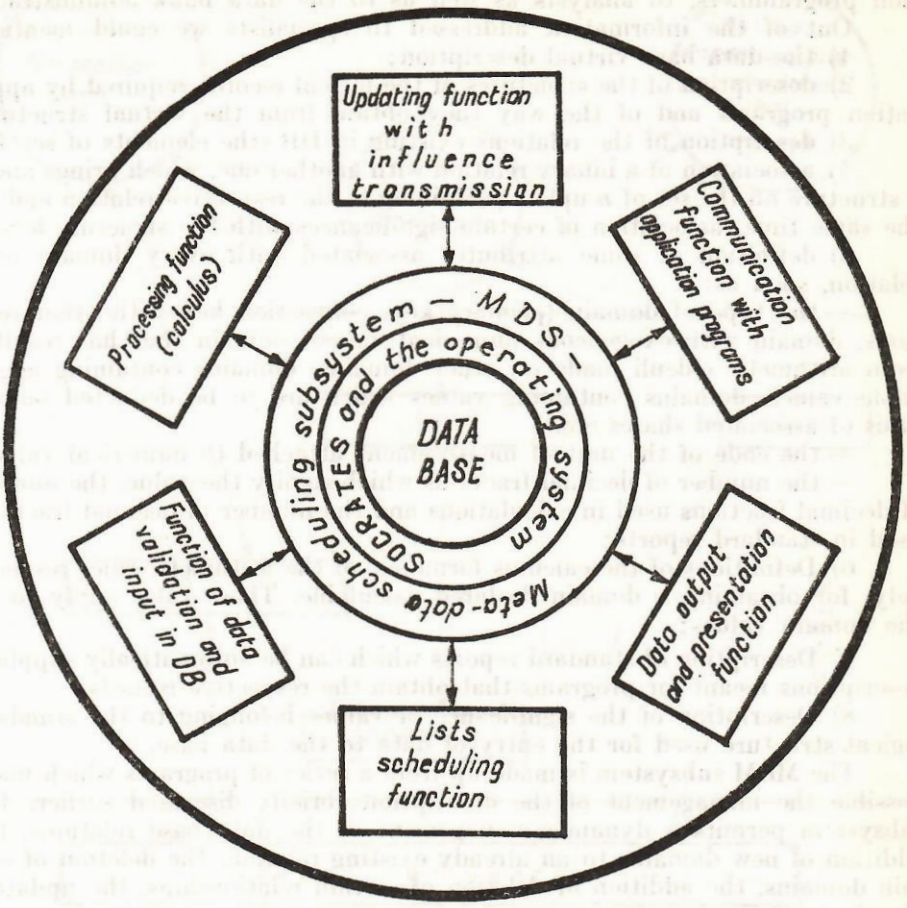

Fig. 2 (Scheduling = management)

1 — achievement of a certain range of data validating operations;

2 — data entry to DB;

3 — unitary management of classified lists;

4 — data updating by automatically ensuring the data base coherence, therefore by transmitting the influences of a change to all data affected by it;

5 — performance of calculations according to the definition of the relation domains;

6 — achievement of interface with programs written in COBOL;

7 — obtaining of standard output records;

These functions can be used both in batch and in a conversation. The present use in batch processing is done by the explicite apply to the needed functions, while in a conversational mode this is done by starting a dialogue directed by SOGER, by interrogating the user until the complete definition of his demand.

The present preoccupations also include that of applying to these functions implicitely by creating a relational enquiry language. The general conception of creating this kind of language based on CODD's works is shown at § 4.3.

4.2.3. The SOCRATES system

The SOCRATES system facilities are the nucleus round which the system is built.

All programs pertaining to MDM and DM subsystems are written by using a combination of SOCRATES enquiry language with ASSIRIS assemblage language.

In this way, SOCRATES facilities are implicitely used by the agency of MDM and DM subsystems components.

4.2.4. Subsystem for the REORGanization and control of DB exploitation efficiency (REORG)

The subsystem permits a control of the performances in data bank exploitation and reorganization when necessities arise.

The subsystem for ensuring data confidentiality and integrity has not been described as it almost exclusively bases on the facilities provided by the SOCRATES system.

4.3. *Orders of the enquiry-designed relational language*

The principle of using relational languages can be acquired by a non-specialist user in three hours [1], in a very short time therefore.

The language, now in design stage, bases itself on the creation of an interpreter written in SOCRATES combined with ASSIRIS which operates with the aid of MDM subsystem.

The user has, as a working tool, a list supplied by MDM, comprising descriptions of the relations existing in the DB together with the coded presentation of the relation domains and the significance of the associated structure levels. As far as the conversational mode is concerned, if the user does not possess this document, he can get it on the display or on the screen recopying printer.

In writing instructions (orders) references will be made to relations and domains through the codes attached to them. We shall further give the structure of the language main orders.

4.3.1. Calculation Order

**C* (relation; list of domains)

— In case the writing form is **C* (relation), then all domains of the indicated relation are covered, and the calculations, resulting from the way domains have been defined, are made:

Calculations can be:

a) aggregations for domains declared calculable;

b) value deductions based on percentage shares;

c) arithmetic calculations by automatically taking into account calculation priorities resulting from calculus formulae;

d) combinations between a, b, c;

— In case one wishes to make calculations for certain domains only, they are specified in the list. Examples:

*C(5;) — calculations are made for all domains of the relation 5;

*C(5; 7, 8, 9, 11) — calculations are made for the domains 7, 8, 9, 11 of the relation 5.

4.3.2. Updating Order

*M (relation, domain: primary key: value T)

— by this order the indicated value is changed and the data base coherence is automatically ensured. Thus, it is ensured the automatic transmission of that change influence to the entire data base.

In case the option T is also mentioned, then the listing of each and every influence is obtained on the display or printer in accordance with the operating mode (conversational or batch).

For example:

*M (4.3,: K :, 1257, T)

4.3.3. Listing Order for the data base information

*L (relation, domain, ..., [; primary key [, binary relation]] : listing conditions)

— the elements "relation, domain" define the contents of the table columns;

— the primary key is indicated in case the listing must begin with a certain line of the relation;

— the binary relation is indicated in case the listing is meant to take a certain structure as far as the classified list is concerned;

— the listing conditions are defined by means of the relational calculus operators indicated by CODD in [1].

5. DATA ORGANIZATION IN THE DATA BASE

As it has resulted from [3] the logical model designed for users is quite simple, due to the fact that they consider the data base to be a set of tables with lines and columns (relations) which they can easily operate.

This simple logical model is supported by a complex, network type structure defined at the virtual level and which ensures the data organization with a minimum degree of redundancy in combination with a minimum response time.

The real way of data organization, the virtual structure included, are elements controlled and managed by the DBA.

We shall further present several suggestive elements for illustrating the conception of creating the virtual structure and of organizing the data, respectively.

Data organization is done in the form of tree-like structures which can be interconnected (network).

Every tree is reflected in the logical model in a n-ary relation of data and a binary relation of associated structure.

Types of trees utilized in data structuring

Figure 3 shows a tree where domains stand for economic indicators and the levels represent two profiles: the departmental and the branch ones.

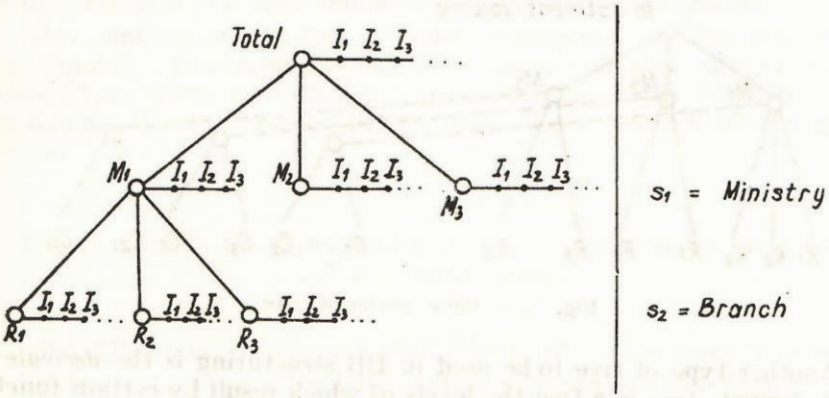

Fig. 3 — Main basic tree

It follows that, within one tree, there are represented the indicators with the same type of share by level.

The allocation of positions on the horizontal line (definition of domains) is made dynamically, the same indicator having the possibility of being several times represented for different planning alternatives or periods.

Ordinary trees used in the data structuring process are called *main basic trees.*

This type of tree is given in Fig. 3.

In certain situations it is necessary to represent trees for which part of the levels has the same significance with the levels of previously defined trees. In such cases, in order to avoid redundancy in the data representation, the new tree must be attached to the previously defined tree by starting with the level whose significance is preserved. Such trees are called *basic anchored trees.*

Figure 4. presents a tree whose level 1 is equivalent to the same level of the tree in Fig. 3.

Figure 5. presents the tree from Fig. 4. as being anchored to the tree from Fig. 3., thus avoiding redundancy in representing the investment volume apportioned by ministry.

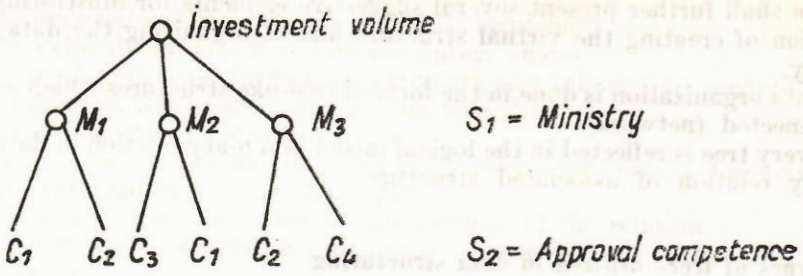

Fig. 4. — Main basic tree

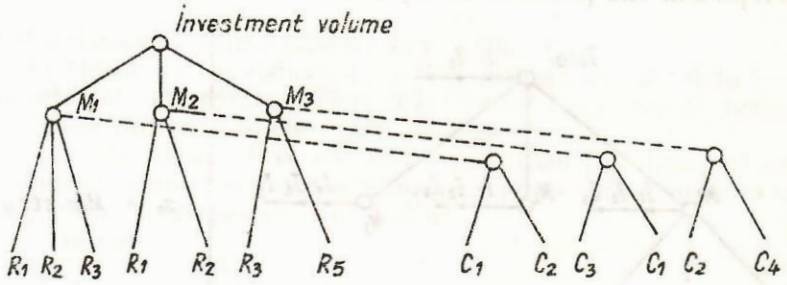

Fig. 5. — Basic anchored tree

Another type of tree to be used in DB structuring is the *derivate tree*. A derivate tree is a tree the levels of which result by certain functions from one side of a basic tree levels.

In its turn, a derivate tree can also be anchored.

In Fig. 6 there is a tree derived (b) from a main basic tree (a). The derivate tree has but one level resulted from the main tree second level.

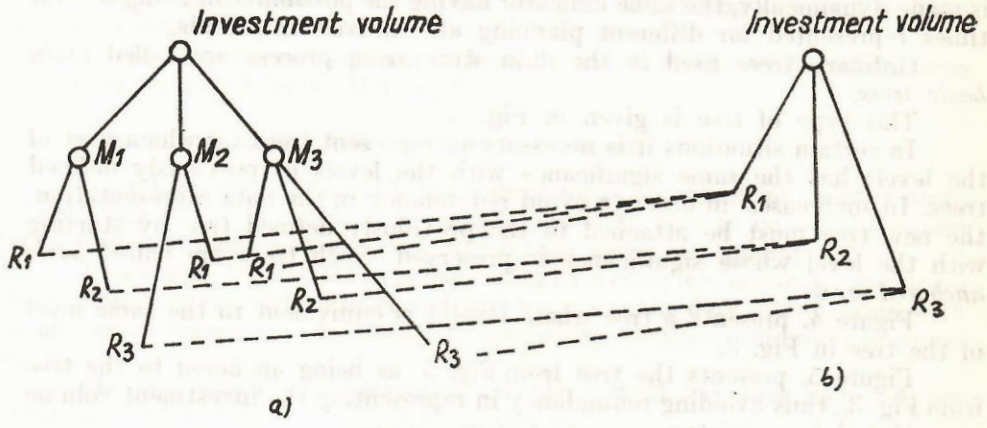

Fig. 6. — Derivate tree

Having the possibility of structuring the data in the above-shown way the DBA must choose the best suited solutions for those cases where it is felt the necessity of introducing new data or of satisfying new users demands.

From the presentation of these types of trees there results the totally different way the DB is regarded by some user in comparison with that of the DBA.

If one were to judge the data organization by the logical model that users have at their disposal, one would reach false conclusions. For example, one of the conclusions would be that of the existence of a high degree of data redundancy, which, in fact, does not happen.

For example, from a logical viewpoint, the equivalent of two trees, a main one and an anchored one, is given by two different relations; if we had no idea of the way data are organized in the form of anchored trees, we would infer the existence of a high redundancy in the data organization.

This combination of a logical model, as simple as possible, placed at the users disposal, simultaneously with ever larger and performative facilities, a desideratum which we consider to achieve through the SOGER system, is the domain of today's preoccupations to develop the data bank management software.

6. UTILIZATION OF **DBMS SOGER** FOR THE DATA BANK MANAGEMENT OF PLANNING (DBP)

SOGER meets, to a great extent, the DBP management requirements. The different data bases of the DBP contain relations in which domains usually stand for indicators belonging to different plan sections and alternatives, to different planning profiles periods. The levels resulted from structuring with the help of binary relations represent plan profiles, various grouping levels for products etc. [10]

The SOGER system simulation facilities are extremely useful in the planning process.

On the basis of the SOGER system facilities in view of organizing the data corresponding to a plan section in the data bank, an analysis is needed for establishing:

a) the necessary output data.

b) the need of filling in the existing lists.

c) the exogeneous indicators.

d) the indicators resulting from calculations and the corresponding calculus formulae.

These elements being established, with a minimum programming effort (a standard tape with input data and, if needed, programs for certain output data) the SOGER system facilities, the way they have already been presented before, can be used in batch or in a conversation.

The change in the planning methodology, the needs for simulation, data structure are nothing else but updatings either of data or at description level (calculus formulae etc.). Programs are not affected due to their working principle through the MDM subsystem; at the most, with a new run they will find other elements of description (other formulae, other level significances etc.) and will act in keeping with the new descriptions.

The SOGER system has been applied to the building of the data base of investment planning, a data base which has been in current use since 1975, to the data base of agricultural planning (the animal and vegetal productions, balances of products for the food industry planning) to the creation of a data base comprising the 1976—1980 five year plan indicators for planning sales, etc.

7. CONCLUSIONS AND DIRECTIONS FOR IMPROVING THE DBMS-SOGER

From the experience gained in the current utilization of the implemented data bases there has resulted the viability of the system design, of the technical way of building it, as well as the adequate performances from the point of view of the response time and the resources employed.

System's further development will be channelled towards the following directions:

1) Removing the degree of redundancy existing in writing generalized programs, components of SOGER system, by a more strict delimitation of certain primitive functions which should combine in order to achieve the existing functions. In this way, programs will be simplified and their performance level will rise.

2) Developing the facilities describing the MDM subsystem, so as to extend the description management both over the data managed by classical system with automated processing and over the data still manually managed.

Thus, an image of all data used in the planning system (in particular will be constructed, with information concerning the way data are being turned to account by emphasizing the existing redundancies etc.

This extension will serve both in improving the planning methodology the information system of planning, respectively, and in designing the new informatic systems based on the DBP utilization.

3) Developing the facilities of interactive utilization by implementing the relational calculus-based enquiry language, but maintaining the present facilities of interactive utilization based on the principle of dialogue directed by SOGER.

The implementation of this language will permit a more concise formulation of the users questions.

4) Parallel to the use of the provisions mentioned at 3, it is passing on to a larger system utilization by non-specialist users.

5) Improving the system's general performances through a deeper analysis of the impact among the three main software elements: generalized programs, the SOCRATES system and the operating system.

REFERENCES

1. CODD E. F.: *A Relational Model of Data for Large Shared Data Banks.* Communications of the ACM, no. 6, 1970.
2. SENKO M. E.: *Data Structures and Data Accessing in Data Base System Past, Present, Future.* IBM System Journal, Vol. 16 no. 3, 1977.
3. UHROWCZIK P. P.: *Data Directory/Directories.* IBM System Journal, Vol. 12, no. 4, 1973
4. ZLOOF M. M.: *Query-by-Example: A Data Base Language.* IBM System Journal, Vol. 16, no. 4, 1977.
5. BLASGEN M. W., ESWARAM K. P.: *Storage and Access in Data Bases.* IBM System Journal, Vol. 16, no. 4, 1977.
6. BARBOSA RUI: *Contribution à l'étude sémantique des bases de données.* Thèse présentée à l'Université scientifique de Grenoble pour obtenir le grade de docteur.
7. DELOBEL C.: *Contribution théorique à la conception et l'évolution d'un système d'informations appliqué à la gestion.* Thése presentée à l'Université de Grenoble.
8. SUNDGREN BO: *Theory of Data Bases.* Petrocelli Information Systems Series, New York, 1975.
9. PESCARU V., CATONA I: *Fișiere, baze și bănci de date.* (Data Files, Bases and Banks), Editura tehnică, Bucharest, 1976.
10. PESCARU V., FIERBINȚEANU G.: *Sistem Informatic integrat pentru planificarea producției* (Integrated System for Agricultural Production Planning). Studii și cercetări de calcul economic și cibernetică economică, no. 4, 1977.
11. PESCARU V., CATONA I., POPESCU C., DRAGOMIR C., UNGHIANU M., USVAT P., VICOLEANU V., DUMITRACHE V.: *Dezvoltarea sistemului de gestiune pentru bănci de date cu noi facilități orientate spre utilizator* (Development of Data Bank Management System with New User-Oriented Facilities), Symposium of Informatics, 1978, Cluj, Romania.
12. PESCARU V., CATONA I.: *Survey of Planning Data Banks.* Modern Trends in Cybernetics and Systems, Springer-Verlag, Berlin, 1977.
13. PESCARU V., DUMITRESCU V., CATONA I., POPESCU C.: *La conception d'une banque de données pour le Comité d'Etat du plan.* Institut de recherche d'informatique et d'automatique, Paris, no. 12, 1973.

DEVELOPPEMENT DES SYSTEMES DE DIRECTION D'UNE BANQUE DE DONNEES PRESENTANT DES FACILITES POUR LES CALCULS CONCERNANT LES STRUCTURES INFORMATIONNELLES ARBORESCENTES

Les auteurs, prenant pour point de départ les principales imperfections des systèmes de gestion pour les banques de données généralisées, soutiennent en tant que solution pour l'élimination de ces défauts (solution plutôt intermédiaire jusqu'à la réalisation de nouveaux SGBD) le prolongement des facilités SGBD généralisées, par des facilités dirigées vers des groupes d'utilisateurs en réalisant des fonctions standard, et rétrécissant dans une certaine mesure la sphère de généralité, mais éliminant les défections.

Cette opinion est soutenue par l'accomplissement pratique, au Centre d'Informatique et de Cybernétique pour le Plan d'un software de gestion pour les bases de données relationnelles nommé SGBD SOGER (SOCRATE augmenté de facilités pour la GEstion d'une base de données Relationnelles utilisant le système généralisé de gestion pour les banques de données SOCRATE), fourni par le C.I.I.-France.

SGBD SOGER se caractérise par les résultats remarquables obtenus dans l'exploitation de la banque de données du plan.

РАЗВИТИЕ БАНКА ДАННЫХ СИСТЕМ УПРАВЛЕНИЯ С ОБЛЕГЧЕНИЕМ ДЛЯ РАСЧЕТОВ КАСАЮЩИХСЯ ДРЕВВИДНОГО РАЗВЁТВЛЕНИЕ ИНФОРМАЦИОННЫХ СТРУКТУР

В настоящей работе, исходя из подчёркивания основных недостатков систем управления для обобщённых банков данных и с целью устранения этих недостатков, авторы выдвигают идею пополнения возможностей систем управления обобщёнными банками данных с возможностями, направленными к группам потребителей, посредством создания некоторых стандартных функций, уменьшая в какой — то мере степень обобщённости, но устраняя таким образом существующие недостатки.

Такой вывод подкреплён практической реализацией в рамках Центра Информатики и Кибернетики по Планированию пакета программ по управлению базами реляционных данных, названного SGBD SOGER и основанного на обобщённой системе управления банками данных (SOCRATF) полученной от C.I.I. — FRANCE.

SGBD SOGER характеризуется хорошими результатами, полученными при го применении в рамках банка данных по планированию.

17. VI. 1978