DATABASE APPROACH FOR MULTIPLE-CRITERIA DECISION SUPPORT SYSTEMS

Mohamed Tawfik Jelassi, Matthias Jarke
Graduate School of Business Administration
New York University

Alain Checroun
Centre d'Etudes et de Recherches en Informatique Appliquee
Universite de Paris-Dauphine
Place du Marechal De Lattre de Tassigny
75775 Paris Cedex 16, France

September 1983
Revised December 1983

Center for Research on Information Systems
Computer Applications and Information Systems Area
Graduate School of Business Administration
New York University

Working Paper Series

CRIS #67

GBA #84-38(CR)

Presented at the First International Summer School on Multiple-Criteria Decision Making: Methods, Applications, and Software, 5-16 September, 1983, Costa Ionica, Sicily, Italy. To appear in a book (Springer-Verlag Publishing Company, 1984).

Table of Contents

Abstract

This paper focuses on data management aspects of computerized decision support systems which use interactive multiple criteria decision methods. In this context, we point out the technical requirements for such systems and the importance of the data management tool to MCDSS.

After a discussion of candidate data models (i. e., relational, hierarchical, and network), we examine the criteria to use in choosing the data model for MCDSS.

In the last part of this paper, we review some database management services which support data definition, data manipulation, and data integrity within the multiple-criteria decision making framework. These services guide us when designing the appropriate architecture for the MCDSS's data component.

1. Introduction

Traditional intuitive methods of decision making are no longer adequate to deal with the complex problems faced by the modern decision maker. The difficulty in addressing these problems is complicated by the interrelations, immediacy, and far-reaching implications of actions taken [11]. Decision support systems have been developed to provide the information and analysis necessary for the decisions. Personal computers, computer networks, databases, color graphics and computer-based models are among the technological developments which are stimulating interest in the use of computers to support decision making [24]. The characteristics of the problems associated with decision support are different from those to which database systems and other computational technologies have normally been applied in the past.

In the particular context of multiple-criteria decision making (MCDM), the data management system should support the following MCDM characteristics:

(1) The explicit knowledge about multiple evaluation criteria within the method (rather than only in the head of the decision-maker), and the change of the relevant criteria over time.

(2) The computational aspect which consists of the search for the efficient solution of the problem at hand; for instance, by computing the objective function, extracting the feasible alternatives and evaluating them.

(3) The frequent interaction between the decision support tool and the end user(s) to assess the objectives, and to check the consistency between the decision making process and the user's preferences [12,13].

(4) The learning process offered by conversational MCDSS, since the decision-maker can express his preferences by weighting the evaluation criteria, making pairwise judgements, or by simply giving an ordinal ranking of alternatives.

The problems are such that it is likely that the decision maker's perception of the problem will change over time and that the inherent nature of the problem itself may change. Hence additional technologies, methodologies, and approaches are needed to make existing database and computational systems more effective in addressing problems of such a nature.

2. Importance of Database Management to MCDSS

Most applications of database systems and computer-based information systems have been aimed at operational control or management control in organizations, and therefore the major

concerns of such systems have been at low levels, dealing primarily with raw data. With multiple criteria decision support systems, data analysis needs are more important. Furthermore, mechanisms must be included for quickly adapting to the changing nature of problems, for assimilating new data series, and for integrating existing models and programs in the effort to save time in responding to a particular decision-maker's request. Hence computational technology as applied to decision support systems needs a new approach, including a better, faster database management system [11].

Altshuler and Plagman [1] have articulated the role of databases in decision support. They consider some of the questions concerning the structure of the database in decision support systems (which they call Corporate Level Systems). In particular, they suggest what data about data should be maintained in these systems in order to help solve such recognized problems of decision support systems as validation and open-ended design.

3. Requirements for MCDSS

In the first part of this section, we will discuss the technical requirements needed to meet the characteristics of the problems that multiple-criteria decision support systems must address. In the second, we will focus on the database requirements for MCDSS.

3.1. Technical Requirements for MCDSS

We recognize the following requirements from the work of Donovan [11] and others:

(1) Data management capability. Because the problems are changing continuously and the answers are needed quickly, the data management capability must have an interactive component that can quickly introduce new data series and validate, protect, and query them. Furthermore, the data associated with these problems is constantly changing and must be obtained from many different sources. Since many of the data series exist under different and potentially incompatible database management systems, and these data series are maintained by people using those systems, it is important to have facilities for integrating data from diverse general purpose database management systems into such a form that a single-user application program may access them.

(2) Analytical capabilities. Coupled with the capabilities of a database, it is important to have sophisticated computational facilities for analyzing the data since the complexity of the problems addressed demand more than raw data. Required capabilities include modeling languages, statistical packages, and other facilities.

(3) Multiple-criteria representation. Depending on the method at hand, the criteria used to evaluate alternatives, and the relationships between these criteria (e.g., weights) are implicit or explicit. The MCDSS must be able to store and present to the user the current system of objectives, as well as apply it to the available alternatives.

(4) Transferability. Because of the short time available for responding to the decision-maker's needs, it is important that the MCDSS is able to build upon existing work, such as existing models or existing programs. It is also necessary to work with data coming from different sources. Hence it is valuable to introduce in an integrated framework any programs, computational aids, or data series that are applicable, even though these programs and data series may currently be operating on seemingly incompatible computer systems.

(5) Reliability, maintainability, and capabilities for incorporating new technologies. Software associated with any decision support system must be reliable, maintainable, and adaptive to changes in technology.

The transferability requirement is important for two reasons: first, as was mentioned above, it allows for the incorporation of existing models, database systems, and other software into an integrated framework quickly; second, it minimizes the need to retrain users of existing systems. Support personnel should be able to operate a computational facility using data management tools and languages with which they are familiar. They would not be required to learn new tools, thus saving time and expense.

Primarily in response to the transferability need outlined above, Donovan's emphasis has been to find techniques for accommodating different database systems and analysis systems in one integrated framework. Rather than force the conversion and transport of application systems to one operating system, Donovan advocates the use of the virtual machine concept, and the networking of virtual machines.

The use of the virtual machine configuration allows different users, working on similar problems, access to the same database. Each user may be familiar with a different analytical system. The data needed by all users may be maintained by one user. In decision-making systems, it is often desirable for different (and often incompatible) application programs (e.g., models) to be able to interact with each other frequently.

3.2. Database Requirements for MCDSS

Among the three components of a multiple-criteria decision support system (i.e., the data, the model, and the dialog management), the data base management system constitutes an important element since it supports the interaction with the decision-maker, as well as with the set of multiple-criteria decision methods used in the decision processes (see fig. 1).

We are interested in investigating how to improve database systems for the goal of supporting decision-making. How can we choose the most appropriate data model for use in MCDSS? Should we base our choice on the operations and integrity constraints, or on the representation of the data? What are the database requirements for MCDSS? Do we need additional database requirements to the 'traditional' ones in building an MCDSS? If so, what is the degree of their importance? Then, based on these requirements, how can we design the MCDSS database component? At which level and up to which degree may a DBMS be used to support the dialog and model management components?

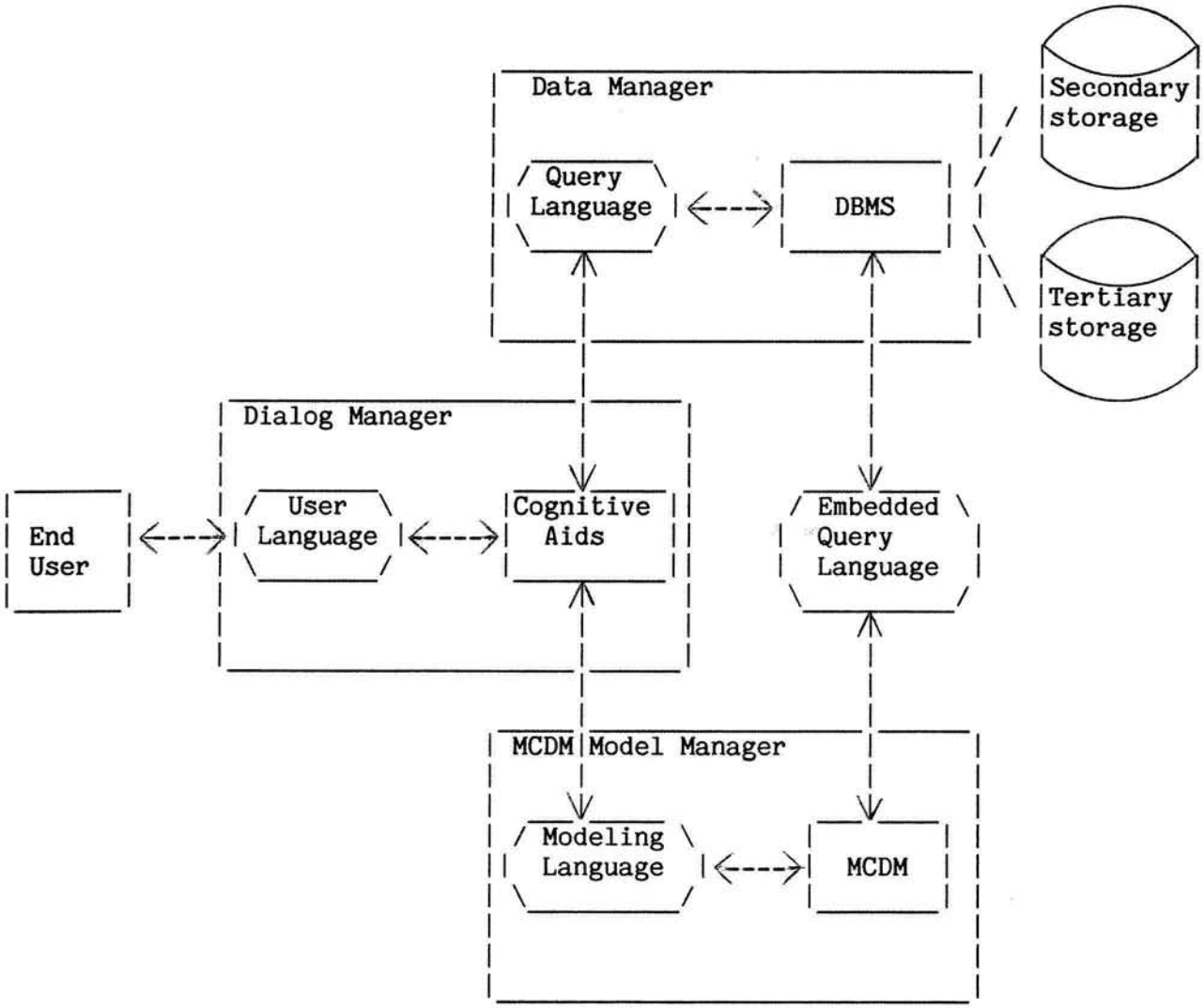


Fig. 1: The MCDSS Architecture (adapted from [26, 28])

Common database requirements for decision support systems, and in particular to MCDSS are:

(1) For the data, the DBMS should support different memory aids such as workspaces for intermediate results, libraries for saving workspaces, links among data, and triggers to remind decision-makers of operations to be performed or of data that should be considered [6]. The amount of data used during the decision making process varies over time, and also varies among decisions and decision-makers. Large volumes of data, coming from internal and/or external sources, are potentially relevant to a decision, but only small volumes are actually needed during the decision process. Since a variety of sources are used, a catalog of data sources would be a valuable aid. Since decision-making insights often come from looking at data in new ways, the DBMS should support establishing relationships and views.

(2) Among the operations needed in a decision making environment, data reduction (or abstraction) from large amounts of data involves subsetting, combination, and aggregating of records and fields in a database. Since minor data errors often vanish in the aggregation process, absolute accuracy may not be required. The most recent data also may not always be required, especially when decisions are based on time series data which cover several years. On the other hand, data reduction does not imply that all data should be aggregates because the decision-maker often wants to examine selected detailed data used to create an aggregate

value. There may also be a hierarchy of decision levels requiring different degrees of aggregation [14].

(3) The decision making process has often a wide time frame. For instance, to perform the forecasting operation (i.e., looking at data from the past and projecting data for the future), several "historical files" and alternative projections for the "current files" are necessary. Random access to all of these data is needed, since insights or questions based on one set of data lead the decision-maker to access data that he did not expect to need, or data that are not related to the data currently being used in terms of storage location or access keys.

(4) Since the users of a MCDSS tend to have low programming skills, the database management system should interface to the user at the external level rather than at the conceptual or internal level [2]. That is, the decision-maker should be free from having to know details of how data are stored and how operations on the database are implemented. In addition, the database component should be integrated with the dialog and model management components.

(5) A key aspect of database performance for interactive MCDSS is its impact on response time. However, good response time is relative to the time frame of the decision and the time constraints and expectations of the decision-makers.

4. Data Models for MCDSS

In building the database management component for MCDSS, there will be an implicit or explicit choice of one or more data models. A data "model" is a method of representing, organizing, storing and handling data in a computer. A data model has three parts [7]:

1. A collection of data structures (lists, tables, relations, hierarchies, and networks). These data structures define the fields and records allowed in the database.

2. A collection of operations that can be applied (usually by the DBMS) to the data structures. The operations define the allowed manipulations of those fields and records. Retrieval, update, combination, and summation are examples of these operations.

3. A collection of integrity rules that define the "legal" states (set of values) or changes of state (operations on values) for the data structures. In other words, the integrity constraints define what fields and record constraints must be preserved by the operations.

The data model for multiple-criteria decision support systems should not be confused with the modeling component of MCDSS. Although the database management component may be used by the modeling component, the data model is a model of data storage and/or operations on storage, whereas the modeling component contains models of decision processes.

There is a rough equivalence among the three 'great' data models (relational, hierarchical, and network) in terms of what information can be represented in each. For a given set of data (objects), there is a representation in any of these models. The

differences among the models are whether the relationships are explicit or implicit, what operations are possible given the representation, and what constraints are placed on the representation.

The logical view of data has been an important issue in recent years in database design, because existing commercial systems are based on one of the three major data models (relational, hierarchical, and network) [8].

4.1. The Relational Data Model

A relational database can be thought of as a collection of tables. The values in each column of a table stem from a common domain such that -- mathematically speaking -- a relation is a subset of the Cartesian product of the domains. Relations can map one-one, one-many, and many-many relationships within the same construct. The relation elements -- rows of the tables -- are usually referred to as tuples. In contrast to the set-theoretic definition of relations, the columns are not identified by their position but by so-called attribute names. Relationships between different tables are usually established through common domains underlying part of their attributes.

Retrieval operations on relations are viewed as the construction of derived relations from existing ones, either by means of a non-procedural description of the desired result or by application of set-oriented algebra operations. While retrieval

operations are powerful set-oriented procedures, altering operations such as insert, delete, and update are typically performed tuple-at-a-time.

Integrity constraints can be defined as predicates over relations much in the same way as retrieval operations. In particular, a subset of attributes whose values describe a tuple uniquely can be defined as a primary key for a relation and this property can be enforced by integrity checks when altering operations are attempted. In summary, the strengths of the relational model are that elegant and powerful theoretical foundations are available for this model, and that it is based on a single concept, the table, which is easily understood by human users.

4.2. The Network Data Model

The basic constructs of a network data model are logical record types represented by rectangles that correspond to entity sets and DBTG-set types represented by arrows that correspond to relationship types in the E/R model [7]. The most important construct in the network model is the set that relates the owner record type to one or more member record types. In case of many-many relationships, one must create a link record type between the two record types. Record occurences correspond to entities, data items to attributes, and set occurences to relationships.

The network model compared with the relational model shows that the former uses predefined acccess paths through set mechanisms, while the latter does not and all possible routes are dynamically materialized; the former specifies certain integrity constraints to the data structure, while the latter only declares them as adjuncts to the data structure. Four major areas of data definition are structure, physical placement, access path, and integrity and privacy constraints [20]. The relational model includes neither physical placement nor access paths, and there is clear separation between structure and constraints, while the network model includes all these four areas of description.

4.3. The Hierarchical Data Model

In this model, the data is represented by a tree structure where each node corresponds to one record (type). The record type at the top of the tree is known as the root, and the elements at the lowest levels, which have no children, are called leaves. No child record type can have more than one parent segment type, whereas in the network structure, as long as the child is in a different set, it can have more than one parent (owner). Therefore, many-many relationaships cannot be handled directly in the hierarchical model. A set type represents the relationship between the owner record type and member record type in a network sense, and it is formally declared; but it is implied in the hierarchical model. No child segment occurrence can exist without

its parent, while can be defined in the network data model by a singular set for which the owner is the "SYSTEM", where the SYSTEM has only one occurrence in the network model.

4.4. Choosing a Data Model

The choice of a data model depends on the users, the decisions, and on the capabilities of the DBMS that could be used. There is a diverse community of users facing a multiple-criteria decision support system. The MCDSS should therefore provide information to all levels of management in a batch or in an interactive mode. The users (including decision-makers at all levels, researchers, and analysts) need standard reports and interaction with the database for data retrieval, modeling, or computation in order to solve structured, semi-structured, or unstructured problems.

Codd [8], McGee [18], Michaels [20], and Ullman [31] outline several factors for database evaluation: being easy to use, easy to understand, easy to manipulate, and easy to implement. These factors should be taken into account while designing the MCDSS's database component.

From these four points of view, the relational model is a very promising approach because of its suitability to non-procedural languages, its separation between structural elements and physical elements maintain data independence, and its

simplicity in data structure and data manipulation. In fact, researchers, decision-makers, and analysts have found that viewing data in the form of a table (relation) is conceptually simple. Multiple-criteria decision support systems have requirements not only for data manipulation but also for facilities for data analysis.

The network model is more complex than the relational because of its rigid structure, and its bundling four areas of data definition into one construct, which make it lose its data independence.

Where many-many relationships exist, the hierarchical structure is even more complex and rigid than the network structure. The user, as Date [10] points out, is forced to devote more time and effort to solving problems that are introduced by the hierarchical data structure and that are not intrinsic to the questions being asked. Of course, hierarchies are a natural way to model truly hierarchical structures from the real world. From an updating operation point of view, the hierarchical model possesses more undesirable properties than the previous two logical data models.

Generally, the relational model is more appealing in the environment of an MCDSS, but we are not suggesting that any one model should be dominant [9, 20]. Because user demands are so diversified, no one model can meet the needs of the large community of users completely for all circumstances [18].

5. A Design for the Database Component of MCDSS

The database requirements for multiple-criteria decision support systems, discussed in section 3.2 of this paper, point out the following key elements which should be considered while designing the database management component of MCDSS:

(1) The characteristics of the data such as the variety of data types, data sources, volume, and level of detail, in addition to the wide time frame.

(2) The operations which support data manipulation, criteria manipulation, subsetting and aggregation.

(3) The interfaces to the other MCDSS components (at the internal level), and to the end user (at the external level).

(4) The system's performance in terms of its response time.

In the following sections, we propose some database management services within the multiple-criteria decision making framework.

5.1 Data Definition Services

(1) A dictionary should exist as a separate component of the database. It will be used to catalog the data in the database by containing descriptions of the criteria, the views, the integrity constraints, and supporting some operations such as adding new

entries, deleting entries, retrieving information on the entries, and maintaining multiple indices (e.g., data name, data type, creation date). Integrating data dictionary operations with other functions is very useful. For instance, deleting an item from the dictionary will result in deleting it from the database.

(2) A special property of MCDSS is the "views capability", i.e., the possibility for the decision-maker to see the data in different ways, as , for example, required by the changes in the evaluation criteria used. A view can be defined as a subset, aggregation, or other combination of data, extracted from the database [16, 21]. We need some operations to determine whether a user is allowed to access customized data structures (databases, records, or fields) or not. A view is not stored in the database; it is only part of the conceptual data model, and that is, a view is defined as a query and implemented using the query language [24]. In order to provide different views of a file, different descriptions may be needed for the same file, and therefore, retained in the database dictionary.

5.2. Data Manipulation Services

(1) The creation, deletion, and update operations are necessary to maintain the database. The query operations are used to manipulate and select records and fields from the database.

(2) Concurrency control [4] determines whether several users can simultaneously access the database (at the field, record, or database level). If sharing is allowed, the DBMS should provide procedures to prevent users from accessing inconsistent data or getting into "deadlock".

(3) Access path optimization [15, 22, 31] and file reorganization [23] are used to increase the performance of the DBMS's operations. To perform these operations, data independence between the programs and the data is a must. Access path optimization consists of choosing the order in which records records are selected (from the storage medium) and minimizing the number of records that have to be searched. File reorganization consists on changing the physical arrangement of records on the storage medium, and/or changing the structure of the database.

(4) In addition to the "classical" operations of a database management system stated above, a "staging process" involving a data extraction facility [5, 17, 33] is required to meet the decision-maker's needs. Its role consists of accessing different kinds of source files, operating on these files, and producing a target file, called the extracted database.

The rationale for a staging process approach is:

(a) That a variety of source databases can be interfaced with the MCDSS database. One example of such a situation, a DSS for supporting container transportation decisions [14] where source

databases are distributed over the world.

(b) Accessing an extracted database is much more efficient than accessing large operational files.

(c) Retrieval of data is easier for a decision-maker from an extracted database than if various source files must be accessed.

The differences between extraction and queries are:

(a) Description of the source files, which may be external to the MCDSS, must be available for the extraction process,

(b) While the answer to a query normally is displayed to the user, the result of an extraction operation is loaded into a target database. The data extraction technique involves aggregating and subsetting from source databases to form an extracted database which is used by the other MCDSS's components, i.e., the dialog and the model management components. With the subsetting operations, the user is able to select fields or records from the source database using a given criteria. With the aggregation operations, he can sum, count, join, or combine several fields or records, using for example one or more common attributes. Aggregation sometimes causes the decision-maker to want to examine the detailed data that were used to create the aggregate. Therefore, it may lead to the need for the inverse operation, disaggregation. If aggregation and subsetting requests are often repeated with few or no modifications , a library of aggregation and subsetting operations is useful.

The staging process [19], represented in fig. 2, consists of four steps:

(a) The reading operation accesses the databases in the tertiary storage (historical data, external data sources, very large transaction processing (TP) data files), and moves the data into the staging process'working area.

(b) The editing operation validates and edits the data, by changing for instance the item names or the data types.

(c) The subsetting and aggregating operations "reduce" the data, as described above.

(d) The loading of the file moves the data from its reduced form to the MCDSS database (the target storage) which contains the model intermediate results, the criterion values, and the raw data.

Another component of the data management tool for multiple-criteria decision support systems (see fig. 2) is the "query language facility". It could be invoked at any time to retrieve quantitative or descriptive information that might be relevant to a particular step in the decision making process. The query operations are invoked by the model and the dialog management components. They produce, as results of the queries, reports or on-line displays. The query language facility must be integrated with the other operations (e.g., insert, delete, update) invoked via the dialog component by the decision-maker. This integration can be realized by using the input-in-context dialog in which the user inputs are always given in the context of the previous output from the MCDSS [24]. The query language facility is used more heavily during the intelligence and choice phases, and used least during design. An end user interface should be part of the query language facility [29] to facilitate

the interaction between the decision-maker and the MCDSS. In this context, we mean by "end user" any user who might, or might not, have a prior experience in the computer field, such as a researcher, an analyst, or a decision-maker at any level of the organization (e.g., a clerk, a manager).

The "generalized view processor" is a key element of the MCDSS's data manager. It plays the following roles:

(a) It processes the translated database transactions coming from the query facility language;

(b) It transforms the raw data coming from the local MCDSS database, through the DBMS, into preprocessed data; and

(3) It interfaces with the staging processor whenever data from the tertiary storage are needed, and with the data dictionary to retrieve the criteria definitions and/or the view definitions (Fig. 2).

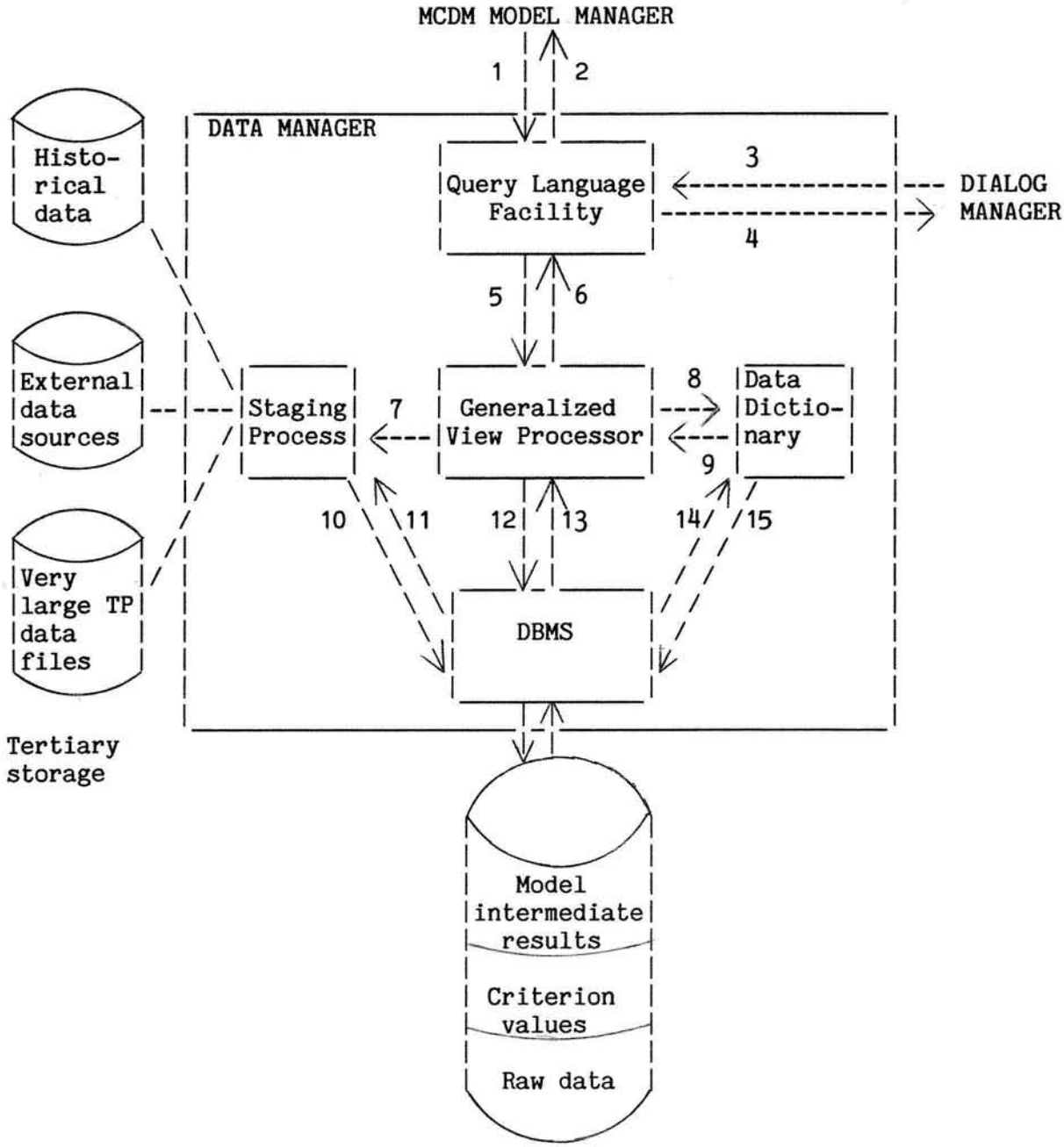
5.3. Data Integrity Services

In this category, we include:

(a) Protection tests [30] which check the decision-maker's password, and control the access level (field, record, or database level),

(b) Recovery operations [32] which restore the database to a consistent state after a hardware or a software failure, and

(c) Integrity checks [30] which consist of checking data for consistency and accuracy in order to validate new entries.



Local MCDM Data-Base

Legend:

1. Query/Insert
2. Data/Definitions
3. Query/Insert/Delete/Update
4. Data/Messages/Definitions
5. Translated transaction
6. Preprocessed data
7. Request
8. Request
9. Criteria definitions/View definitions
10. Load
11. Unload
12. Data base transaction
13. Raw data
14. Request
15. Data definitions/Integrity constrainrts

Fig. 2: The Data Management Component of the MCDSS

6. Conclusion

Designing a database-oriented MCDSS represents a new challenge, which requires integrating computer technology, management science, and organizational theory.

A decision-maker needs two capabilities: data management and modeling. The difference between a conventional DBMS and a DBMS that can support decision-making process is that the latter has to incorporate the capability to handle models and on-line interfaces to databases [3, 25].

The design criterion for such a comprehensive system is that it must handle the interfaces among the users [27], databases, and models. In addition, user involvement and knowledge of the multiple criteria decision making application to be developed are required in implementing a successful MCDSS.

References

1. Altshuler, G., and Plagman, B., "User/System Interface within the Context of an Integrated Corporate Database", in Proceedings AFIPS, 1974 National Computer Conference, Vol. 43, AFIPS Press, Montvale, N.J., pp. 27-33.

2. ANSI/X3/SPARC (Standards Planning and Requirements Committee), "Interim Report from the Study Group on Database Management Systems", FDT (Bulletin of ACM-SIGMOD), Vol. 7, No. 2, February 1975.

3. Bennett, J. L., "Building Decision Support Systems", Englewood Cliffs, N. J.: Prentice-Hall, 1983.

4. Bernstein, P. A., and Goodman, N., "Concurrency Control in Distributed Database Systems", ACM Computing Surveys, Vol. 13, No. 2, 1981.

5. Carlson, E. D., "Using Large Databases for Interactive Problem Solving", in Proceedings of the International Conference on Very Large Databases, ACM, 1975, pp. 499-501.

6. Clemons, E. K., "Database Design for Decision Support", in Proceedings of 14th Hawaii International Conference of System Science, 1981.

7. "CODASYL Database Task Group Report, 1971", CODASYL, Association for Computing Machinery, New York, April 1971.

8. Codd, E. F., "Data Models in Database Management", in Proceedings of the Workshop on Data Abstraction Databases, and Conceptual Modeling, ACM Order Number 474800, 1980, pp. 112-114.

9. Codd, E. F., "A Relational Model of Data for Large Shared Data Banks", Communications of the Association for Computing Machinery, Vol. 13, No. 6, June 1970, pp. 377-387.

10. Date, C. J., "An Introduction to Database Systems", Third Edition Reading, Mass.: Addison-Wesley, 1981.

11. Donovan, J. J., "Database System Approach to Management Decision Support", ACM Transactions on Database Systems, Vol. 1, No. 4, December 1976, pp. 344-369.

12. Jacquet-Lagrèze, E., and Shakun, M. F., "Decision Support Systems for Semi-Structured Buying Decisions", to appear in the European Journal of Operational Research (1983).

13. Jacquet-Lagrèze, E., and Siskos, J., "Assesssing a Set of Additive Utility Functions for Multicriteria Decision-Making: the UTA Method", European Journal of Operational Research, Vol. 10, No. 2, June 1982, Amsterdam, Netherlands: North-Holland, pp. 151-164.

14. Jarke, M., "Developing Decision Support Systems: A container Management Example", International Journal of Policy Analysis and Information Systems, Special issue on Decision Support Systems, Vol. 6, No. 4, December 1982, pp. 351-372.

15. Jarke, M., and Koch, J., "A Survey of Query Optimization in Centralized Database Systems", Working Paper, November 1982, Center for Research on Information Systems, Computer Applications and Information Systems Area, Graduate School of Business Administration, New York University.

16. Jarke, M., and Vassiliou, Y., "Coupling Expert Systems with Database Management Systems", in: Artificial Intelligence: Applications for Business, W. Reitman, (ed.), Ablex, Norwood, N. J., to appear, 1984.

17. Lee, D. T., "Database-Oriented Decision Support Systems", in Proceedings AFIPS, 1983, National Computer Conference, AFIPS Press, Montvale, N.J., pp. 453-465.

18. McGee, W. C., "On User Criteria For Data Model Evaluation", ACM Transactions on Database Systems, Vol. 1, No. 4, December 1976, pp. 370-387.

19. Methlie, L. B., "Data Management for Decision Support Systems", Database, Vol. 12, No. 1-2, Fall 1980, pp. 40-46.

20. Michaels, A. S., Mittman, B., and Carlson, C. R., "A Comparison of the Relational and CODASYL Approaches to Data-Base Management", ACM Computing Surveys, Vol. 8, No. 1, March 1976, pp. 125-150.

21. Rosenthal, A., and Reiner, D., "Querying Relational Views of Networks", in Proceedings of the IEEE COMPSAC Conference, Chicago, Illinois, 1982.

22. Selinger, P., Astrahan, M. M., Chamberlin, D. D., Lorie, P. A., and Price, T. G., "Access Path Selection in a Relational Database Management System", in Proceedings of the ACM-SIGMOD Internaational Conference on the Management of Data, Boston, 1979, pp. 23-34.

23. Sockut, G. H., and Goldberg, R. P., "Database Reorganization - Principles and Practice", Computing Surveys, 11, 1979, pp. 371-396.

24. Sprague, R. H., Jr. and Carlson, E. D., "Building Effective Decision Support Systems", Englewood Cliffs, N. J.: Prentice-Hall, 1982.

25. Stohr, E. A., and Tanniru, M. R., "A Database for Operations Research Models", International Journal of Policy Analysis and Information Systems, Vol. 4, No. 1, 1980, pp. 105-121.

26. Stohr, E. A., "DSS for Cooperative Decision Making", in Proceedings of the NATO Conference on Database and Decision Support Systems, 1981, Amsterdam, Netherlands: North-Holland.

27. Stohr, E. A., and White, N. H., "User Interfaces for Decision Support Systems: An Overview", International Journal of Policy Analysis and Information Systems, Special issue on Decision Support Systems, Vol. 6, No. 4, December 1982, pp. 393-423.

28. Stohr, E. A., and Ginzberg, M. J., "Decision Support Systems: Issues and Perspectives", in: Decision Support Systems, M. J. Ginzberg, W. Reitman, and E. A. Stohr, (eds.), Amsterdam, Netherlands: North-Holland, 1982, pp. 9-31.

29. Stohr, E. A., and White, N. H., "Languages for Decision Support Systems", to appear in: Management and Office Information Systems, S. K. Chang, (ed.), Plenum Press: New York, 1983.

30. Stonebraker, M., "Implementation of Integrity Constraints and Views by Query Modification", in Proceedings of ACM-SIGMOD International Conference on the Management of Data, San Jose CA, 14-16 May 1975, pp. 65-77.

31. Ullman, J. D., "Principles of Database Systems", Computer Science Press, Second edition, 1982.

32. Verhofstad, J. S. M., "Recovery Techniques for Database Systems", Computing Surveys 10, (1978), pp. 167-197.

33. Walker, A., "On Retrieval from a Small Version of a Large Database", in Proceedings of the Sixth International Conference on Very Large Databases, Montreal, 1980, pp. 47-54.