DESIGNING A GENERALIZED MULTIPLE CRITERIA DECISION SUPPORT SYSTEM

**Mohamed Tawfik Jelassi, Matthias Jarke
and Edward A. Stohr**

June 1984

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

Working Paper Series

CRIS #78

GBA #84-60

Presented at the VIth International Conference on Multiple-Criteria Decision Making (MCDM), June 4-8, 1984, Cleveland, Ohio. To appear in: **Lecture Notes in Economics and Mathematical Systems**, Springer-Verlag (1985).

Abstract

Decision support systems are of many kinds depending on the models and techniques employed in them. Multiple criteria decision making techniques constitute an important class of DSS with unique software requirements. This paper stresses the importance of interactive MCDM methods since these facilitate learning through all stages of the decision making process. We first describe some features of Multiple Criteria Decision Support Systems (MCDSSs) that distinguish them from classical DSSs. We then outline a software architecture for a MCDSS which has three basic components: a Dialog Manager, an MCDM Model Manager, and a Data Manager. We describe the interactions that occur between these three software components in an integrated MCDSS and outline a design for the Data Manager which is based on a concept of levels of data abstraction.

1. INTRODUCTION

Micro-computers, databases, color graphics, and computer based models are among the technological developments which are stimulating interest in Computerized Decision Support Systems (DSS) [Sprague and Carlson, 1982]. DSS are used to support decision making activities in a variety of:

(1) situations (structured, semi-structured, or unstructured) [Mintzberg et al., 1976; Keen and Scott-Morton, 1978],

(2) managerial levels within the organization (operational, tactical, and strategic) [Anthony, 1965],

(3) phases of the decision-making process (intelligence, design, and choice) [Simon, 1960], and

(4) decision styles of the decision-makers (rational, empirical, intuitive, or political) [Huber, 1980].

Various definitions have been suggested for decision support (e.g., [Benbasat, 1977], [Joyce and Oliver, 1977], [Sprague and Watson, 1979], and [Keen, 1980]). We will use the one given by Ginzberg and Stohr [1982] who define a DSS as "a computer-based information system used to support decision making activities in situations where it is not possible or not desirable to have an automated system to perform the entire decision process". Based on this definition, an MCDSS can be considered as a 'specific' type of system within the broad family of DSSs. The major software components of an MCDSS are shown below in Figure 1. Even though they include the same basic components (i.e., data handling, model handling, and dialog handling systems), MCDSSs are distinguished from other DSSs by the kind of mathematical models they use and, therefore, the nature of the decision-making environment they are intended to deal with.

While it is customary to view models as the focal point of decision support, the emerging perspective is shifting the focus to the database as the foundation for an MCDSS. This has raised the following questions: How must database systems be modified in order to support multiple-criteria decision-making? In other words, what are the database requirements for MCDSS? At which level and to which degree may the Data Manager be used to support the Dialog Manager and the MCDM Model Manager?

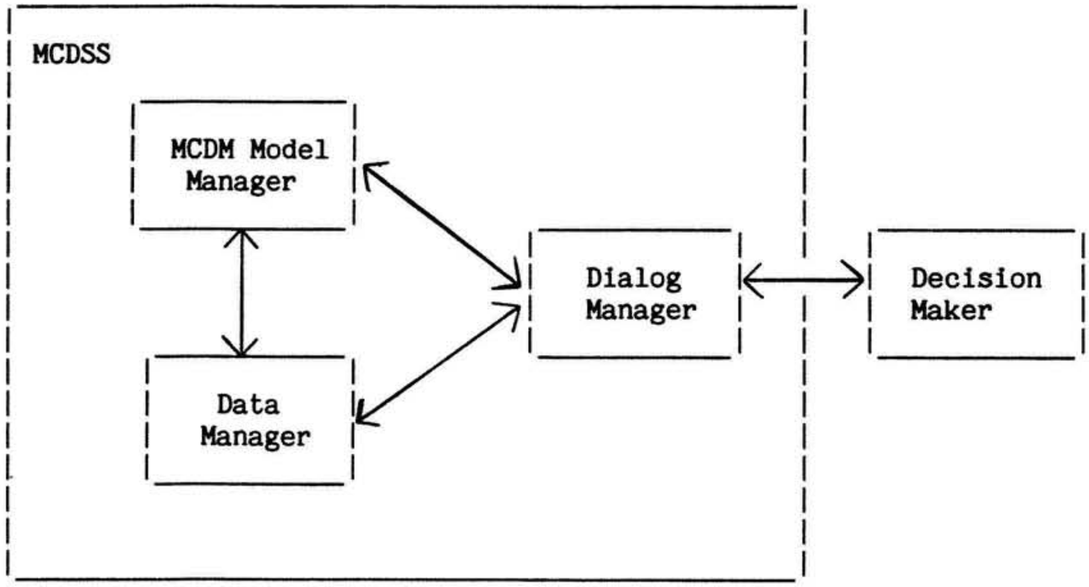

Figure 1: The Dialog - Data - Model Paradigm
[Bonczek et al., 1981]

2. THE MCDSS INGREDIENTS

This section describes some features of the MCDM environment that must be satisfied by an MCDSS. We then outline our proposed architecture for an MCDSS by describing each of the three components of Figure 1, and detailing their interaction in solving an MCDM problem.

The following features of MCDM models distinguish MCDSSs from classical decision support systems:

(1) They allow analysis of several criteria at once. These criteria may be either quantifiable (or 'objective') such as price, weight, etc., or non-quantifiable (or 'subjective') such as comfort, aesthetics, quality of service, etc. [Zeleny, 1982]. The criteria may also be conflicting [Shakun, 1981], i.e., the improvement or achievement of one criterion can be accomplished only at the expense of another.

(2) They allow subjective evaluation of the criteria and alternatives by weighting, rating, or ranking them.

(3) They support interactive decision-making especially for semi-structured problems with large sets of alternatives. The reactions of the decision-maker to the results obtained in any one stage of this process influence the criteria definition, and a-priori preferences, as much as the problem formulation.

(4) They can be used for preference analysis of alternatives either to find a satisfactory compromise, or to determine a finite ordered subset of good alternatives.

(5) They may use a variety of multiple-criteria techniques to compute the efficient solution (e.g., ordinal comparisons [Geoffrion, 1970; Geoffrion et al., 1972; Dyer, 1973], preference modelling [Vincke, 1981], aspiration levels, pairwise alternative comparisons [Zionts and Wallenius, 1983], aggregation (of the criteria) and/or disaggregation (of a wholistic preference) [Jacquet-Lagrèze, 1979], consistency checks [Bowman, 1963], implicit utility functions [Fishburn, 1970; Keeney and Raiffa, 1976] constructed by automatic adaptation to the user's answers [Jacquet-Lagrèze and Siskos, 1982], outranking relations [Roy, 1973; Roy, 1974; Roy, 1977], discretization algorithms [Bernabeu and Winkels, 1981], concordance/discordance approaches [Roy, 1978; Bernabeu, 1980], linear multicriteria systems [Zionts and Wallenius, 1976; Winkels, 1981]).

(6) By using the multicriteria techniques (feature (5)) in an interactive way (feature (3)), the role of the MCDM models is to identify and explain the inconsistencies between the two perceptions (that of the decision-maker on the one hand, and that of the MCDSS as an expert/informer on the other) in a learning process.

The MCDSS must support the above processes and provide easy access to a wide range of standard MCDM methods as enumerated in (5) above.

Several surveys of computer based MCDM that are relevant to the above discussion have appeared in the MCDM literature [Wallenius, 1975; Zeleny, 1976; Hwang and Masud, 1979; Despontin and Spronk, 1979; Despontin et al., 1980; Vincke, 1982; Zionts, 1982].

2.1. The Dialog Manager

Interactive MCDM methods (as distinguished from batch processing ones) offer the capability of supporting the interaction with the end user at any stage of the decision making process. In an MCDSS architecture, this feature is made possible through a "Dialog Manager" (see Figure 1). This manages the two-way flow of information between the decision-makers and the Model Manager and Data Manager components of the system. The Dialog Manager provides a number of facilities including:

(1) device independence (the physical characteristics of devices and communications protocols are transparent to users and systems builders),

(2) screen management,

(3) graphics - input and output, and

(4) menu - drivers.

These features can be used in an MCDM context to build a system that will allow all of the different kinds of user-system dialog that were described above.

2.2 The Model Manager

An important component of the MCDSS Model Manager is the model base which provides facilities for the definition storage, retrieval and execution of a wide range of procedures and models. These may be permanent or ad-hoc models and may range in size from very small to very large. Generally a "library" of such procedures will be made available to users and they will also be given the opportunity to "specialize" models for their own use or to build entirely new models. A major function of the Model Manager is to ensure that models execute in a logically consistent manner and that they are properly matched with the most current data. Another major function is to facilitate sensitivity analysis by providing mechanisms for multiple executions of models and (via links with the Data Manager) for the storage, retrieval, display and analysis of results [Stohr and Tanniru, 1980].

2.3. The Data Manager

This component is used to manage all the flows of data within the system. As can be seen from Figure 4, it is envisioned that a full-function data base management system (DBMS) will be employed in this component. Also different sources of data such as data gathered by the MCDSS builders from external sources and internal data processing data files will be accommodated. A major objective of this paper is to describe a logical structure for the data to be managed by the Data Manager to maximize its usefulness to both system builders and end users (decision-makers).

2.3.1. MCDM Data Requirements

MCDM models need a variety of data (e.g., criteria and alternative definitions and values, intermediate and/or final results). Figure 2 summarizes the data interaction between the MCDM Model Manager on one hand and the Data Manager and Dialog Manager on the other hand. A distinction is made between input and output data, its module source and module destination, and the database and data dictionary components of the Data Manager.

From a design point of view, the following criteria, proposed in [Sprague and Watson, 1975], should be taken into account when defining the data structures and operations of MCDM models:

(1) There must exist a mechanism whereby models can extract data from a database. Thus a model is a 'user' of the database, which suggests that a model must have available to it some language with which it can direct information retrieval.

(2) There must be a command language that allows convenient, direct access to the database and that allows execution of available models.

(3) The system must be flexible in terms of modifying the procedural knowledge inherent in the available models and in modifying the ways in which models can be used. This facility permits the system's modeling capabilities to evolve.

2.3.2. MCDM Data Structures

A variety of data structures, including vectors, arrays, and records [Tremblay and Sorenson, 1976], are utilized by computerized MCDM models. Examples of implemented MCDM methods using such data structures are ELECTRE [Bernabeu, 1980], MIRACLE [Bernabeu and Winkels, 1981], and PREFCALC [Euro-Decision, 1983]. Data are physically stored as files and maintained by the 'classical' file management system. A different approach consists of utilizing the general theory of arrays [More, 1973] to extend the concept of database to contain not only data of the decision-making support system but also the data manipulation rules of models [Orman, 1982].

MCDM Model Manager	Input inf.	Output inf.	Raw data	Elaborate data	Data Manager		Dialog Manager
					Database	Data Dictionary	
Criteria names	+		+			-	+
Criteria definitions	+		+			+	-
Criteria values	+		+	-	+		-
Alternatives names	+		+			-	+
Alternatives definitions	+		+			+	
Alternatives values	+		+		+		
Model Messages (notification of completion or interrupt)		+	+				+
Information (parameters / data) requests		+	+		-		+
Messages (interrupt)/ Parameter collection	+		+				+
Commands	+		+				+
Intermediate model results	-	+		+	+		+
Final model results		+		+	-		+

Legend
+ General case
- Exception

Figure 2: Data Interaction between the MCDM Model Manager and the Dialog and Data Managers

Data should be presented to a user in a form suited to both the user's skills and the application required. The choice of a data structure to be supported at the user level (external or conceptual [ANSI/X3/SPARC, 1975]) critically affects many components of the system. It also dictates the design of the corresponding data manipulation language (DML), because each DML operation must be defined in terms of its effect on these data structures [Date, 1981]. Thus the question "Which data structure(s) and associated operators should the system support?" is a crucial one.

Since both the relational data model [Codd, 1970] and multiple-criteria decision making methods typically present data to their users in the form of a table, where rows correspond to entities and columns correspond to properties, tables seem to be the most appealing MCDM data structure. The tabular representation is simple and universally understood. It has the following properties [Codd, 1971]:

P1: It is column-homogeneous; in other words, in any selected column the items are all of the same kind, whereas items in different columns need not be of the same kind;

P2: Each item is a simple number or a character string (thus, for example, if we look at the item in any specified row and any specified column, we do not find a set of numbers or a repeating group).

For database tables, we add three more properties:

P3: All rows of a table must be distinct (duplicate rows are not allowed);

P4: The ordering of rows within a table is immaterial;

P5: The columns of a table are assigned distinct names and the ordering of columns within a table is immaterial.

As a result of P3, each row can be uniquely identified (or addressed) by its content. Property P2 is the only normalization requirement for a relational database. The second, third, and fourth normal forms all represent improved qualities in record structures [Kent, 1983]. These normal forms, however, are not required for implementation of a relational database [Sandberg, 1981].

Property P4 and property P5 do not hold in typical MCDM data structures since an ordering of the alternative outcomes of the decision-making process is desired by the end user. Each row (or "record") of the table corresponds to a "tuple" of the relation, and a column (or "field" of the record) corresponds to an "attribute". The entries in a given column belong to the set of values constituting the domain underlying that column.

Figure 3 shows an example of a database relation (table) in an MCDM context (see Section 3.1 for a more detailed explanation of the application). The table contains six quantitative criteria characterizing ten reference cars. Here the column Make-Model forms the "primary key" for the table (uniquely identifies rows). Similar tables can be imagined for other application areas such as home-buying, computer-selection, etc. For later use, we illustrate the accepted notation for describing relational tables. Thus, the CARS table is defined by:

CARS (MAKE-SERIES, MAX-SPEED, FUEL-CONSUMPTION, ...)

where CARS is the name of the table and MAKE-SERIES, MAX-SPEED, FUEL-CONSUMPTION, etc. are column names.

CARS

Make-Model	Maximal Speed (km)	Consumption in town (lt/100km)	Consumption at 120km/hr (lt/100km)	Horse Power (CV)	Space (m^2)	Price (Francs)
Peugeot 505 GR	173	11.4	10.01	10	7.88	49,500
Opel Record 2000 LS	176	12.3	10.48	11	7.96	46,700
Citroën Visa Super E	142	8.2	7.30	5	5.65	32,100
VW Golf 1300 GLS	148	10.5	9.61	7	6.15	39,150
Citroën CX2400 Pallas	178	14.5	11.05	13	8.06	64,700
Mercedes 230	180	13.6	10.40	13	8.47	75,700
BMW 520	182	12.7	12.26	11	7.81	68,593
Volvo 244 DL	145	14.3	12.95	11	8.38	55,000
Peugeot 104 ZS	161	8.6	8.42	7	5.11	35,200
Citroën Dyane	117	7.2	6.75	3	5.81	24,800

Figure 3: Illustration of Table Structure (Source: "L'Action Automobile et Touristique: Spécial Salon", No. 238, October 1980)

2.4 Summary of the MCDSS Architecture

The three basic components (i.e., the Data, the Model, and the Dialog Managers) interact with each other as described in the following paragraphs. Together, they provide the basis for building an integrated MCDSS. Figure 4 represents a detailed architecture of the MCDSS, based on previous work [Jelassi et al., 1983].

The Model Manager draws input values from the database and produces results while the Dialog Manager passes model results (intermediate or final) and classical database management operations (i.e., query, insert, delete, update) to and from the user. In this paper, however, we will be primarily concerned with the Data Manager component. The MCDM Model Manager and the Dialog Manager interact with the Data Manager through the "Query Language Facility". The latter component can be invoked at any time to retrieve quantitative or descriptive information that might be relevant to a particular step in the decision making process. It plays the following roles:

(1) From the MCDM Model Manager, it receives queries about the data needed as inputs for the model(s), and requests the insertion of intermediate results in the database. Then, it translates these transactions and transmits them to the DBMS through the "Generalized View Processor".

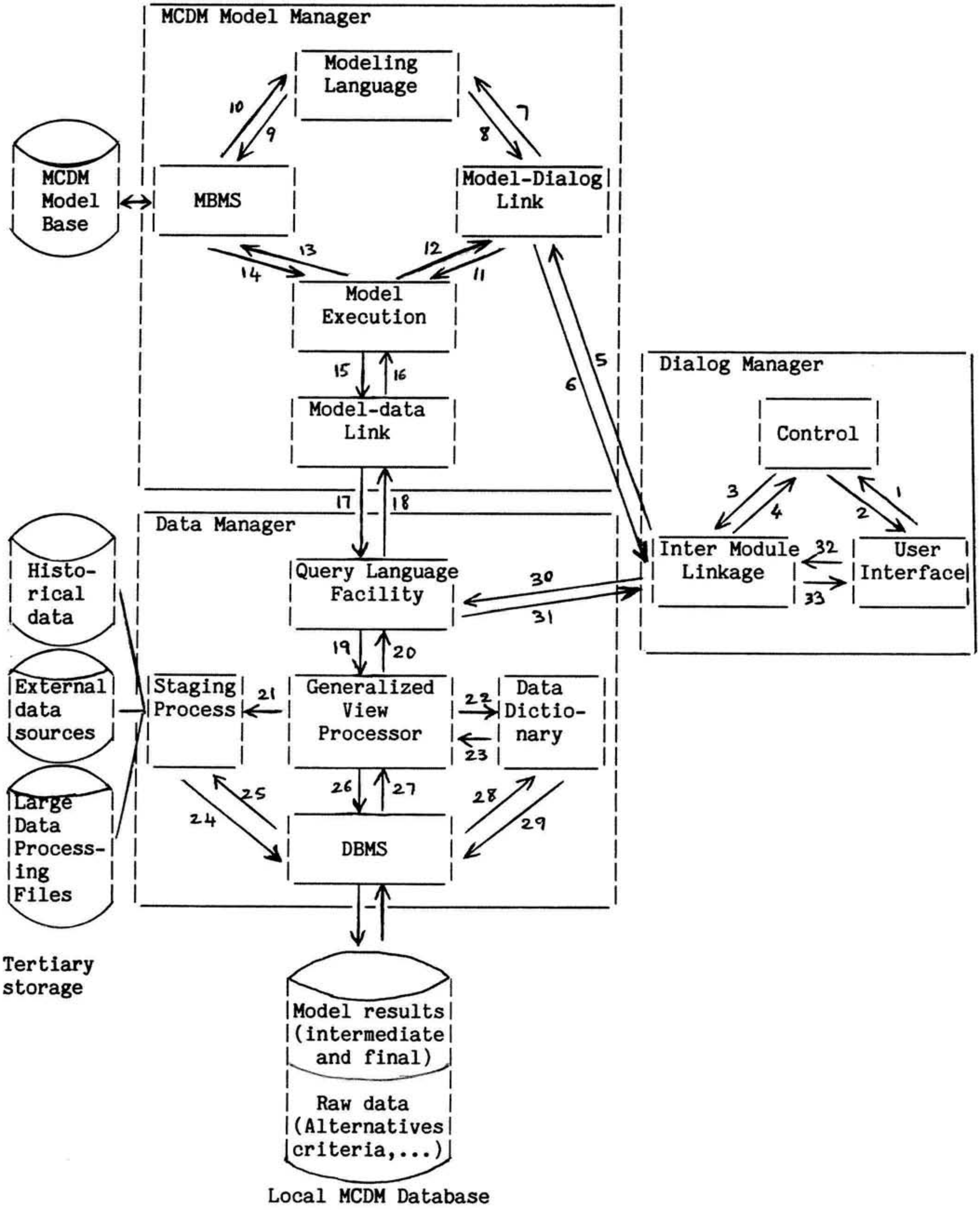

Figure 4: A Detailed Architecture for the MCDSS

LEGEND:

1. Commands / Messages / Information (Parameters, Data) collection
2. Messages / Information (Parameters, Data) request / Data
3. Invocation of model/data transaction
4. Data / Messages / Definitions / Requests
5. Model base transaction / Messages (Interrupt) / Parameter collection
6. Messages (Notification of completion or interrupt) / Parameter request
7. Modeling transaction
8. Messages / Input request
9. Invocation of model base operations (Creation of a new model / Update of an existing one)
10. Parameter request to redefine, restructure, or create a model / Messages (Notification of completion or interrupt)
11. Model execution transaction
12. Messages / Parameter request from the user
13. Invocation of a model
14. Retrieval of the model
15. Data request from the database / Load MCDM results (intermediate/final)
16. Data collection (Model results - intermediate/final - ; Raw data - criterion/alternative values -)
17. Query/Insert requests
18. Data/Definitions collection
19. Translated transaction
20. Preprocessed data
21. Data retrieval request
22. Data (criteria, alternative, model variable, view) definition request
23. Data definition collection
24. Load data from tertiay storage into the local MCDM database
25. Unload data (raw data, model results)
26. Database transaction
27. Raw data / model results collection
28. Data definition request
29. Data definitions / Integrity constraints collection
30. Invocation of a DBMS operation (Query / Insert / Delete / Update)
31. Raw data / Model results / Messages / Definitions
32. Commands / Messages / Information (Parameters, Data) collection
33. Messages / Information (Parameters, Data) request / Data

Figure 4: A Detailed Architecture for the MCDSS (continued)

(2) It passes back to the MCDM Model Manager the answers to its queries, providing raw data, criteria/alternative values, or model results retrieved from the database.

(3) From the Dialog Manager, the Query Language Facility receives requests to manage and maintain the database (i.e., query, insert, delete, and/or update the data values).

(4) In return, it passes to the Dialog Manager different kinds of data (raw, elaborate), and from the data dictionary, definitions of the criteria, the views and the data stored in the MCDM database.

The Model Execution Component is a working storage module where the execution takes place. It draws input values from the MCDM database and requests parameters from the end user through the Dialog Manager. Storing both the intermediate and final results of the MCDM processes in the database allows the establishment of linkages between the models. This component passes the results when any model needs input generated as output from a different model. It also allows presentation of the results to the end user through the Data-Dialog Link. Therefore, no results are passed directly from the MCDM Model Manager to the end user.

The Model-Dialog Link supports the interaction between the user and the MCDM Model Manager. The user determines the sequence of the exchanged messages which is not fixed for each use of the model. This is a particularly important feature of the interactive modeling in a multiple criteria decision support system.

3. A DATA MODEL FOR THE MCDSS

In this section, a practical example will be used to demonstrate the organization and extraction of MCDM-related data. First, the problem is described. Then, the organization of data at several levels of abstraction is illustrated. These levels parallel the decision-making stages involved in MCDM Method selection, designation of criteria for the evaluation of possible choices, restriction of the set of choices and execution of the resulting model.

3.1 Presentation of the Problem

A leading European car manufacturer is interested in building an evolutive decision support system intended to help make better and faster decisions concerning:

(1) The launch of a new model (or version) on the market and its potential consequences vis-a-vis the clientele;

(2) The withdrawal of an existing model (or version) from the production lines;

(3) The development of a "health diagnostic"; in other words, the assessment of the suitability of a model (or a version) from a marketing point of view (e.g., sales volume, quality of the distribution, maintenance facilities, competitive vendors), as well as from a technical point of view (i.e., the performance of a given product);

(4) The modification of the existing strategy according to the results of the "health diagnostic" (e.g., reduction of the production volume, review of some components of a product, introduction of new technologies, definition of a new commercial approach).

The mathematical models used in the car MCDSS involve MCDM and data analysis (namely factor and regression analyses) methods.

3.2 Abstraction Levels for MCDM Data

This subsection mirrors the conceptual representation of the MCDM data by using several "levels of abstraction". This notion was first introduced by [Bonczek et al., 1981] for "network" database models. In our approach, which utilizes the relational data model, levels of abstraction are achieved by having table identifiers and/or column identifiers stored as data values in tables at a conceptually higher level of abstraction. Thus "metadata" concerning an application area can be stored in a hierarchical fashion and manipulated by the database system itself. This approach requires certain modifications to traditional relational database operations since the objects manipulated are not raw data but rather tables containing either raw data or a mixture of raw data and information about other tables.

Abstraction levels for the MCDM problem can be illustrated through the car example introduced above. The set of criteria available for use by decision-makers is stored in the CARS table (Figure 3) and might include the price, maximum speed, fuel consumption, space, horse power, color, and weight of each car. These criteria are attributes of cars represented by the names of columns in the table. On the other hand, "criterion" is a concept that has instantiations such as "price", "fuel consumption", etc. that may vary from application to application and amongst different decision-makers within the same application or for the same user over time. To record this "higher level" information, we can store it in a CRITERIA table as shown in Figure 5. Here the domain of the "Criterion" column is the set of names of columns of database tables while the notation "cars.price", "cars.max-speed",... signifies that PRICE and MAX-SPEED are columns in the CARS table, etc.

CRITERIA

CRITERION	APPLICATION	USER
cars.price	auto-selection	Smith
cars.max-speed	auto-selection	Jones
cars.price	auto-selection	Jones
.	.	.
.	.	.
home.price	home-buying	Adams
.	.	.
.	.	.

Figure 5: "CRITERIA" Table -- at Higher Level of Abstraction than the "CARS" Table of Figure 3

We can represent the different abstraction levels of the example stated above as shown in Figure 6. This depicts a number of database tables (full outlines) together with an indication of the kinds of data stored in their record occurrences (dashed outlines).

Level 1 contains the basic data of an application. In this case, the CARS table contains data values '7.88 square meters', '10.01 liters per 100 kilometers at 120 km per hour', '173 km/hr', '49,500 French Francs', etc., associated with the car 'Peugeot 205', which is a basic unit of choice. Hence, level 1 is the lowest level of abstraction within the database.

The CRIT-ALT table at level 2 is a dynamically generated "virtual table" or "view" containing data extracted from level 1. This data is then passed to the MCDM method. The table may contain a variable number of columns and rows depending (respectively) on a user's choice of criteria and restriction of the set of alternatives to be considered. Information concerning these user decisions is stored in the CRITERIA and ALTERNATIVE tables at level 3. The CRITERIA table has already been explained. The ALTERNATIVE table records the use of categories to simplify the decision-making problem. Thus a user may wish to consider "compact" cars or "station-wagons" in car-buying or "2 bed-room apartments" in a home-buying application.

Table APP-METH at level 4 serves a similar book-keeping function as it records information related to a user's choice of a method (e.g., UTA) in an application area (e.g., car-buying). The most abstract concepts in this example are the 'APPLICATION' and 'METHOD' tables that constitute level 5. These tables describe, respectively, the applications and MCDM algorithms that are currently available in the MCDSS. Thus, among other things, this level provides the system with some "self-knowledge" of what it can and cannot do.

This "multilevel network" corresponds to the "generalization abstraction" for relational databases as proposed in [Smith and Smith, 1977]. It is to be noted that we have relaxed our definition of "level of abstraction" somewhat to include a more natural, problem-oriented view of hierarchical levels as well as the "meta data" view defined above.

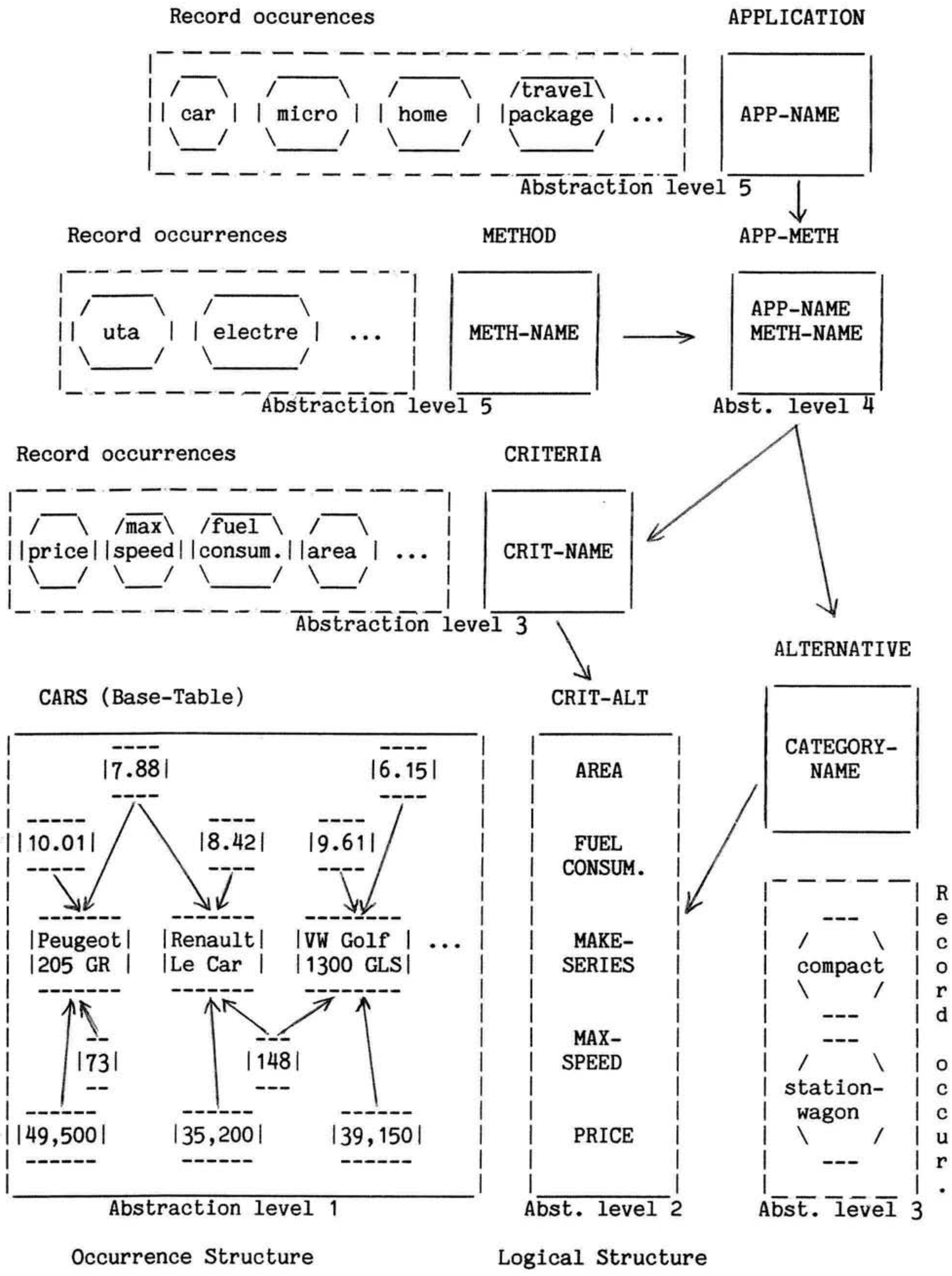

Figure 6: The Abstraction Levels

A more detailed explanation of the tables in Figure 6 is now provided. Columns containing metadata are indicated by a caret symbol. Values stored in these columns are names associated with other database objects. In the case of virtual tables the column names may be variables (indicated by an asterisk). This allows queries and programs to be written that will manipulate the data in the tables before the actual content of the tables is known. This facility is used in our example to allow the system to process different applications such as car-buying and home-buying without additional programming.

Level 1: Basic Data for the Different Applications

CARS (MAKE-SERIES, CAR-CLASS, PRICE, MAX-SPEED, FUEL-CONSUMPTION, etc.)

Tables containing data for other applications (e.g., HOMES, etc.) also exist at this level.

Level 2: Virtual Table - Real Data

```
                  *             *                 *
 CRIT-ALT (CHOICE-IDENT, {CRITERION-1, .. , CRITERION-n})
```

This is created during a particular work-session, either interactively by the user, or if previously defined, it is generated automatically from the data in the CRITERIA and ALTERNATIVE tables (see below). The data in this table are passed (perhaps in matrix form) to the MCDM method. The columns names are variables that are bound to column-name values dynamically during the session. The braces indicate that a variable length list of criteria may be chosen by a user. For the car application, CHOICE-IDENT is 'MAKE-SERIES', while CRITERION-1 might be instantiated to 'MAX-SPEED', etc. This descriptive information will also be passed to the MCDM method.

Level 3: Mixture of Meta and Real Data

```
                    ^         ^
     CRITERIA (CRIT-NAME, APP-NAME, USER-NAME, DATE, etc.)
```

This is generated as users use the system. It contains a history of the lists of criteria chosen by different users. It helps the system automatically retrieve details of the last model used by a given decision-maker.

```
                      ^
ALTERNATIVE (CATEGORY-NAME, CATEGORY-SIZE, CREATION-DATE,
                                 ^          ^
     RESTRICTION-DEFINITION, APP-NAME, METH-NAME, USER-NAME, DATE)
```

This table contains information defining restrictions on the set of choices that may be required by a particular user within a given application using a particular method. CATEGORY-NAME is a system-builder or user-supplied name for a subset of rows in a 'base table' such as CARS, or HOMES, etc. RESTRICTION-DEFINITION contains character strings expressing logical restrictions on the rows in the base data tables of level 1. For example, if a user wishes to consider only compact cars the string might be "where car-class = 'compact' ". These two level 3 tables are used by the "Generalized View Processor" of Figure 4.

Level 4: Mixture of Meta and Real Data

```
              ^          ^
APP-METH (APP-NAME, METH-NAME, USER, TIME, etc.)
```

This table contains information concerning the actual use of a method in problem-solving within a particular application area. It is automatically updated by the system during decision-making sessions. It is available for user inquiry and is used by the system, for example, in automatically restoring the state of the 'last' problem solving session for a given user.

Level 5: Mixture of Meta and Real Data

```
            ^                             ^
METHOD (METH-NAME, METH-DESCRIPTION, DATA-TRANSFORM, AUTHOR, etc.)
```

This relation contains information on MCDM methods. It is maintained and updated by the system builder/maintainer. METH-NAME has domain consisting of the names of the executable load modules of the MCDM methods. Similarly, DATA-TRANSFORM has as domain the set of names of executable procedures which will transform the data in the CRIT-ALT relation into formats suitable for use by the MCDM methods. This relation is used to supply the user with information on which methods are available and the applications for which they are suitable. It is used by the system when users wish to choose an MCDM method to solve their application.

```
                ^                             ^
APPLICATION (APP-NAME, APP-DESCRIPTION, CHOICE-IDENT,
             ORIGINATOR, etc.)
```

This table is the main repository of information concerning applications that have been defined in the MCDSS. APP-NAME has as domain the set of names of tables containing the 'real' data of the applications (i.e., CARS, HOMES, etc.). CHOICE-IDENT contains the name of the column which identifies specific cars or homes, etc. This relation is used to supply the user with information on available applications, and by the system when a method is associated with an application chosen by the user.

The sequence of actions that take place when a user signs-on to the system is as follows:

1. Choose an MCDM application
2. Select an MCDM method
3. Select the category of alternatives of interest
4. Select a subset of criteria
5. Retrieve the data values
6. Execute the algorithm of the MCDM method.

The decision-maker performs the steps 1, 3, and 4. Step 2 can be executed either by the user, or by the model manager (namely, the Model Base Management System (MBMS)). In the latter case, an internal algorithm associates one decision-making application to one (or many) MCDM methods. Referring to Figure 4, step 5 is performed by the Data Manager using the "Query Language Facility", the "Generalized View Processor", and when necessary, the "Data Staging Processor". Lastly, the Model Manager executes the algorithm of the MCDM method chosen at step 2. This operation takes place in the "Model Execution" module.

It is to be noted that these operations involve "stepping down through" all five levels of abstraction. A simple example of the usefulness of the metadata is provided by step 1 where stored knowledge (in the APPLICATION table) of the name of the level 1 data table can be used by the system to automatically open the corresponding file. Similarly, in steps 2 and 6, the METHOD table provides information concerning the MCDM load modules and their associated data conversion procedures. This allows the system to load and activate these modules automatically.

In order to make use of the metadata during MCDSS development, several extensions to the usual relational database languages are required. As an example, suppose that an application developer wishes to provide the system with the ability to automatically restore the problem state at the end of the last user session. Steps 3, 4 and 5 above could be programmed using an SQL-like language [Date, 1981] as shown below. Note that language key-words are capitalized while the names of database objects are in small letters. Also that braces around the variables indicate that columns of relations are to be transformed into character strings or lists of character strings. These transformations are required because relational languages return results as database tables whereas queries are expressed in terms of an entirely different data type, namely character strings. We will frequently wish to retrieve metadata from relations at a higher abstract level and use it to direct queries on lower level tables.

(1) Select Category of Alternatives

Assume that the user is 'Smith' and that this value is stored in a variable 'U-Name'. The last application used by Smith is found from the query:

```
A = SELECT App-name
    FROM   Alternative
    WHERE  User-name = U-Name
      AND  Date = SELECT Max(Date)
                  FROM   Alternative
                  WHERE  User-name = U-Name
```

If Smith's last application was Car-buying, then this query will result in a relation A having one row and one column with value 'CARS'.

```
R = SELECT Restriction-definition
    FROM   Alternative
    WHERE  App-name = {A}
      AND  User-name = U-Name
```

The new relation R will contain a logical condition defining the subset of rows of the CARS table with which Smith was working. For example, R might be: Car-class = 'Compact'.

(2) Select Criteria

```
C = SELECT Crit-name
    FROM   Criteria
    WHERE  App-name = {A}
      AND  User-name = U-Name
```

This will form a new table, C, containing the names of the criteria used by Smith in the car-buying application.

(3) Retrieve the Data Values

```
I = SELECT Choice-ident
    FROM   Application
    WHERE  App-name = {A};
```

Table I contains the name of the column in the base table which serves as the identifier for the alternatives from which the user must choose. In our example, this is MAKE-SERIES.

```
ALTERNATIVE = SELECT {I}, {C}
              FROM   {A}
              WHERE  {R};
```

Finally, this generates the virtual table ALTERNATIVE, which contains the information required by the MCDM method.

Conclusion

In this paper, we have outlined some of the requirements for a generalized software system suitable for MCDM applications (an MCDSS) and listed some of the special requirements that differentiate this from a normal DSS. We described major software components -- a Dialog Manager a Model Manager and a Data Manager. The interaction between these three components was described and a fairly detailed design for the Data Manager Component was developed. A major contribution of the paper is the application of the notion of database "levels of abstraction" to produce a database design specialized for use by MCDSS developers and end users. The design calls for the storage of metadata which essentially mirrors the formulation of MCDM problems in terms of methods, applications, criteria for choice, sets of alternatives and so on. Examples of operations that utilized this metadata and cross the boundaries from higher, more abstract, levels in the database to more concrete data-oriented levels were given.

It is the authors' intention to pursue this line of research and to develop an MCDSS that will reduce the development time for MCDM builders and provide a fully user-friendly interface to users. Such general purpose software should be able to support a wide range of MCDM methods, lend itself to the development of new methods and applications, and be easily transportable among different computer systems.

References

[ANSI/X3/SPARC, 1975]

Study Group on Database Management Systems: Interim Report. FDT, Vol. 7, No. 2, Bulletin of ACM SIGFIDET, February 1975.

[Anthony, 1965]

Anthony, R. N., "Planning and Control Systems: A Framework for Analysis", Harvard University Graduate School of Business Administration, Studies in Management Control, Cambridge, Mass., 1965.

[Benbasat, 1977]

Benbasat, I., "Cognitive Style Considerations in DSS Design", Database, Vol. 8, No. 3, Winter 1977.

[Bernabeu, 1980]

Bernabeu, Y. A., "Manuel d'Utilisation du Programme ELECTRE III", Cahier du Laboratoire d'Analyse et Modélisation de Systèmes pour l'Aide à la Décision (LAMSADE) No. 35, University of Paris Dauphine, Paris, July 1980.

[Bernabeu and Winkels, 1981]

Bernabeu, Y. A., and Winkels, H. M., "MIRACLE: Une Methode Interactive de Reduction, D'Amelioration et de Classification Evolutive", Document du Laboratoire d'Analyse et Modélisation de Systemes pour l'Aide ã la Décision (LAMSADE) No. 16, University of Paris Dauphine, Paris, June 1981.

[Bonczek et al., 1981]

Bonczek, R. H., Holsapple, C. W., and Whinston, A. B., "Foundations of Decision Support Systems", Academic Press, 1981.

[Bowman, 1963]

Bowman, E. H., "Consistency and Optimality in Managerial Decision Making", Management Science, Vol. 9, 1963, pp. 310-321.

[Codd, 1970]

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.

[Codd, 1971]

Codd, E. F., "Normalized Database Structure: A Brief Tutorial", Proceedings of ACM-SIGFIDET Workshop on Data Description, Access, and Control, San Diego, California, 1971, pp. 1-17.

[Date, 1981]

Date, C. J., "An Introduction to Database Systems", Third Edition, Reading, Massachusetts: Addison-Wesley Publishing Company, 1981.

[Despontin and Spronk, 1979]

Despontin, M., and Spronk, J., "Comparison and Evaluation of Multiple Criteria Decision Models: First Results of an International Investigation", Erasmus University of Rotterdam, October 1979.

[Despontin et al., 1980]

Despontin, M., Moscarola, J., and Spronk, J., (eds.), "A User-Oriented Listing of Multiple Criteria Decision Methods", Free University of Brussels, V.U.B., Center for Statistics and Operation Research, Report CSOOTW, 152, December 1980.

[Dyer, 1973]

Dyer, J. S., "A Time-Sharing Computer Program for the Solution of the Multiple Criteria Problem", Management Science, Vol. 19, No. 12, August 1973, pp. 1379-1383.

[Euro-Decision, 1983]

PREFCALC: User Manual, 1983, Euro-Decision inc., Buc, France.

[Fishburn, 1970]

Fishburn, P. C., "Utility Theory for Decision Making", John Wiley and Sons, 1970.

[Geoffrion, 1970]

Geoffrion, A. M., "Vector Maximal Decomposition Programming", Working Paper No. 164, Western Science Institute, University of California, Los Angeles, September 1970.

[Geoffrion et al., 1972]

Geoffrion, A. M., Dyer, J. S., and Feinberg, A., "An Interactive Approach for Multicriterion Optimization with an Application to the Operation of an Academic Department", Management Science, Vol. 19, No. 4, December 1972, pp. 357-368.

[Ginzberg and Stohr, 1982]

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

[Huber, 1980]

Huber, G. P., "Organizational Science Contributions to the Design of DSS", in: G. Fick, and R. H. Sprague, (eds.), "Decision Support Systems: Issues and Challenges", Pergamon Press, 1980, pp. 45-55.

[Hwang and Masud, 1979]

Hwang, C. L., and Masud, A. S., "Multiple Objective Decision Making Methods and Applications: A State-of-the-Art Survey", Lecture Notes in Economics and Mathematical Systems, No. 164, Springer-Verlag, 1979.

[Jacquet-Lagreze, 1979]

Jacquet-Lagrèze, E., "De la Logique d'Aggrégation de Critères à une Logique d'Aggrégation-Disaggrégation de Préférences et de jugements", Cahier de l'ISMEA,: Série Sciences de Gestion, Tome XIII, No. 4-5-6, 1979, pp. 839-859.

[Jacquet-Lagrèze and Siskos, 1982]

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

[Jelassi et al., 1983]

Jelassi, M. T., Jarke, M., and Checroun, A., "Database Approach for Multiple Criteria Decision Support Systems", First International Summer School on Multiple Criteria Decision Making: Methods, Applications, and Software, September 5-16, 1983, Costa Ionica (Sicily). To appear in a book, Springer-Verlag (1984). NYU Working Paper Series, CRIS 67, GBA 84-24 (CR).

[Joyce and Oliver, 1977]

Joyce, J. D., and Oliver, N. N., "Impacts of a Relational Information System on Industrial Decision", Database, Vol. 8, No. 3, Winter 1977, pp. 15-21.

[Keen and Scott-Morton, 1978]

Keen, P. G. W., and Scott-Morton, M. S., "Decision Support Systems: An Organizational Perspective", Addison-Wesley Series on Decision Support, Reading, Massachusetts: Addison-Wesley Publishing Company, 1978.

[Keen, 1980]

Keen, P. G. W., "Decision Support Systems: Translating Analytic Techniques into Useful Tools", Sloan Management Review, Vol. 21, No. 2, 1980.

[Keeney and Raiffa, 1976]

Keeney, R. L., and Raiffa, H., "Decisions with Multiple Objectives: Preferences and Value Tradeoffs", John Wiley and Sons, 1976.

[Methlie, 1980]

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

[Mintzberg et al., 1976]

Mintzberg, H., Raisinghani, D., and Theoret, A., "The Structure of 'Unstructured' Decision Processes", Administrative Science Quarterly, Vol. 21, June 1976, pp. 246-275.

[More, 1973]

More, T. Jr., "Notes on the Development of a Theory of Arrays", IBM Philadelphia Scientific Center Report No. 320-3016, March 1973.

[Orman, 1982]

Orman, L., "An Array Theoretic Specification Environment for the Design of Decision Support Systems", International Journal of Policy Analysis and Information Systems, Special issue on Decision Support Systems, Vol. 6, No. 4, December, 1982, pp. 373-391.

[Roy, 1973]

Roy, B., "How Outranking Relation Helps Multiple criteria Decision Making", in: J. L. Cochrane and M. Zeleny, (eds.), "Multiple Criteria Decision Making", University of South Carolina Press, Columbia, South Carolina, 1973, pp. 179-201.

[Roy, 1974]

Roy, B., "Critères Multiples et Modélisation des Préférences: L'Apport des Relations de Surclassement", Revue d'Economie Politique, Vol. 84, No. 1, January-February 1974, pp. 1-44.

[Roy, 1977]

Roy, B., "Partial Preference Analysis and Decision-Aid: The Fuzzy Outranking Relation Concept", in: D. E. Bell, R. E. Keeney, and H. Raiffa, (eds.), "Conflicting Objectives in Decision Making", John Wiley and Sons, Chichester, 1977, pp. 40-74.

[Roy, 1978]

Roy, B., "ELECTRE III: Un Algorithme de Classement Fondé sur une Représentation Floue des Préférences en Présence de Critères Multiples", Cahier du Centre d'Etudes et de Recherche Opérationnelle, Vol. 20, No. 1, 1978.

[Sandberg, 1981]

Sandberg, G., "A Primer on Relational Database Concepts", IBM Systems Journal, Vol. 20, No. 1, 1981, pp. 23-40.

[Shakun, 1981]

Shakun, M. F., "Formalizing Conflict Resolution in policy Making", International Journal of General Systems, Vol. 7, No. 3, 1981.

[Simon, 1960]

Simon, H., "The New Science of Management Decision", Harper and Row, Publishers, New York, 1960.

[Smith and Smith, 1977]

Smith, J. M., and Smith, D. C. P., "Database Abstractions: Aggregation and Generalization", ACM Transactions on Database Systems, Vol. 2, No. 2, June 1977, pp. 105-133.

[Sprague and Watson, 1975]

Sprague, R. H. Jr., and Watson, H. J., "MIS Concepts - Part II", Journal of System Management, Vol. 26, No. 2, 1975.

[Sprague and Watson, 1979]

Sprague, R. H. Jr., and Watson, H. J., "Bit by Bit: Toward Decision Support Systems", California Management Review, Vol. 12, No. 1, 1979.

[Sprague and Carlson, 1982]

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

[Stohr and Tanniru, 1980]

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, March 1980, pp. 105-121.

[Tremblay and Sorenson, 1976]

Tremblay, J. P., and Sorenson, P. G., "An Introduction to Data Structures with Apllications", Mc Graw-Hill Computer Science Series, Mc Graw-Hill Book Company, 1976.

[Vincke, 1981]

Vincke, P., "Preference Modelling: A Survey and an Experiment", Operational Research'81, J. P. Brans, (ed.), IFORS, 1981, Amsterdam, Netherlands: North-Holland, pp. 341-354.

[Vincke, 1982]

Vincke, P., "Presentation et Analyse de Neuf Methodes Multicriteres Interactives", Cahier du Laboratoire d'Analyse et Modelisation de Systemes pour l'Aide a la Decision (LAMSADE) No. 42, University of Paris Dauphine, Paris, December 1982.

[Wallenius, 1975]

Wallenius, J., "Comparative Evaluation of Some Interactive Approaches to Multicriterion Optimization", Management Science, Vol. 21, No. 12, August, 1975, pp. 1387-1396.

[Winkels, 1981]

Winkels, H. M., "A Flexible Decision Aid Method for Linear Multicriteria Systems", Presented at the Task force Meeting on "Multiobjective and Stochastic Optimization", International Institute for Applied Systems Analysis (IIASA), Laxenburg, Austria, 30 November - 4 December, 1981.

[Zeleny, 1976]

Zeleny, M., (ed.), "Multiple Criteria Decision Making", Proceedings, Kyoto, 1975, Lecture Notes in Economics and Mathematical Systems, No. 123, Springer-Verlag, New York, 1976.

[Zeleny, 1982]

Zeleny, M., "Multiple Criteria Decision Making", Mc Graw Hill Book Company, New York, 1982.

[Zionts and Wallenius, 1976]

Zionts, S., and Wallenius, J., "An Interactive Programming Method for Solving the Multiple Criteria Problem", Management Science, Vol. 22, No. 6, February 1976, pp. 652-663.

[Zionts, 1982]

Zionts, S., "Multiple Criteria Decision Making: An Overview and several Approaches", Working Paper No. 454, January 1982, School of Management, State University of New York at Buffalo.

[Zionts and Wallenius, 1983]

Zionts, S., and Wallenius, J., "An Interactive Multiple Objective Linear Programming Method for a Class of Underlying Nonlinear Utility Functions", Management Science, Vol. 29, No. 5, May 1983, pp. 519-529.