The Design of a Knowledge-Based Guidance System for an Intelligent Multiple Objective Decision Support System (IMODSS)

J. Lu*, M.A. Quaddus*, K.L.Poh** and R. Williams**
*Graduate School of Business, Curtin University of Technology
***School of Information Systems, Curtin University of Technology
Perth, WA 6845, Australia
** Department of Industrial and Systems Engineering
National University of Singapore, 119260

luj@cbs.curtin.edu.au
quaddusm@gsb.curtin.edu.au
isepohkl@leonis.nus.edu.sg
williamsr@cbs.curtin.edu.au

Abstract

This paper describes a project that extends the multiple objective decision support system (MODSS) by offering knowledge-based guidance to an intelligent multiple objective decision support system (IMODSS). This IMODSS integrates expert system (ES), multiple objective decision-making (MODM) methodologies, graphical user interface (GUI) and decision support systems (DSS) technologies. This IMODSS uses an expert system shell CLIPS to build a knowledge base to guide the decision-makers (DMs) to select the most suitable MODM method(s) from the MODM methodology base in order to solve their particular decision problems. This IMODSS has been implemented and tested. This paper mainly discusses the design and implementation of the knowledge-based intelligent guidance subsystem in IMODSS.

Keywords

Decision Support Systems, Knowledge-Based Systems, Systems Design, Multiple Objective Decision-Making

INTRODUCTION

Over the last two decades multiple objective decision-making (MODM) methods have been widely researched because of the theoretical challenge and their practical applications to a wide variety of problems. A large number of MODM methods thus have been developed. However, the literature has shown that some methods are more suitable and efficient than others in the solution of some decision problems by some DMs. Hence a MODSS should preferably contain a sufficient number of MODM methods in its methodology base so that the DMs have the option to choose any method. However, to utilize the methodology base effectively, a MODSS should be designed to have the capability of guiding the DMs to select and use the most suitable MODM methods from the methodology base (Bui and Sivasankaran, 1988; Pinson and Moraitis, 1996; Poh 1998). A knowledge-based intelligent subsystem is thus necessary to achieve this aim.

In a recent paper we have provided a framework for a knowledge based multiple objectives decision support system (KB-MODSS) (Lu, Quaddus and Williams, 1999). This framework has been implemented and tested as a graphical user interface based multiple objective decision support system prototype, called IMODSS. The prototype, with its database,

methodology base of MODMs, knowledge based system and GUI techniques, can select and apply the most appropriate method to support the DMs to solve various MODM problems. One of the advantages of this system is to allow the DMs to select and use the most appropriate methods for each particular decision problem. Another important advantage is its ability to resolve complex problems that could not otherwise be solved with a single MODM, or to allow the DMs to get solutions from different methods. The structure of the IMODSS is shown in Figure1.

The development of the knowledge-based intelligent subsystem is the most important part of IMODSS. This main system, the methodology base containing seven MODM methods and other subsystems of IMODSS have been implemented using Delphi. The knowledge-based intelligent guidance system is designed to represent human expertise in the specific domains of selecting the suitable method and has been implemented by using an expert system shell CLIPS which has been embedded in Delphi. This paper provides the structure of embedding intelligent guidance within MODSS, outlines the specific guidance framework for selecting a suitable MODM method and presents the design process of the method selection knowledge base (MSKB) and the MSKB intelligent guidance system in IMODSS.

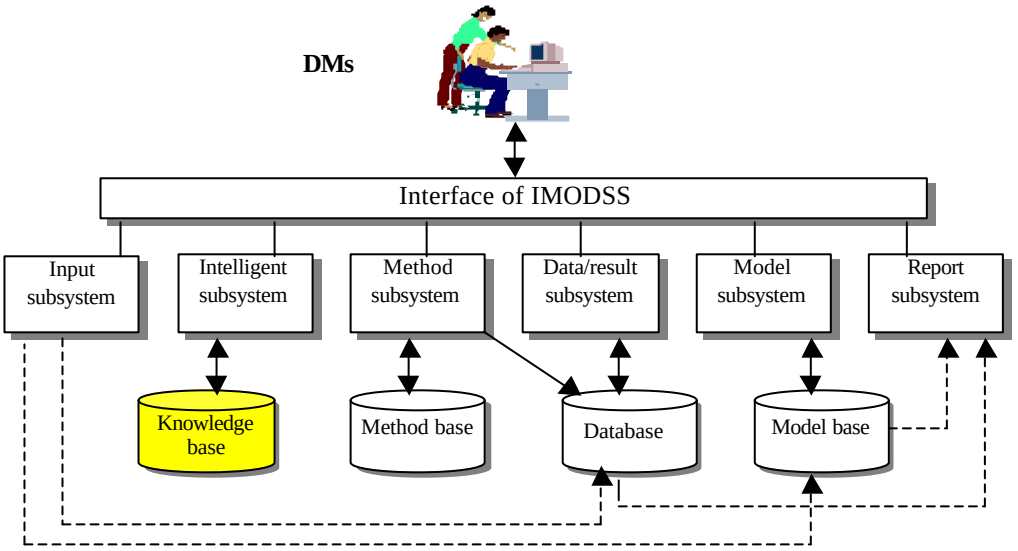

Figure 1: System architecture of IMODSS

ACQUISITION AND IDENTIFICATION OF KNOWLEDGE

Process of MODM Knowledge Acquisition

The knowledge acquisition is the process of capturing the expert's knowledge about the domain into the system. This process includes two main phases: the collection of data and the representation of the facts constituting the expertise in the system's knowledge base (Klein and Methlie, 1995). The collected data should be identified before keeping it in a knowledge base. We use the following steps to identify the knowledge of MODM methods (Figure 2).

- Method identification: identifying a number of traditional and popular MODM methods based on literature review such as Hwang and Masud (1979), Poh, Quaddus and Chin (1995) to build a MODM methodology base.
- Validity recognition: a number of validities are recognized. They are conceptual validity, logical validity, experimental validity and operational validity.

- Methods comparison: comparing all methods included in this system through different points of view and classes.
- Characteristics and concepts identification: the characteristics and concepts of the MODM methods are identified.
- Selection of the type of knowledge representation: there are four main types of knowledge representation schemes in a knowledge base: production rules, semantic nets, frames and logic. We used types of production rules.

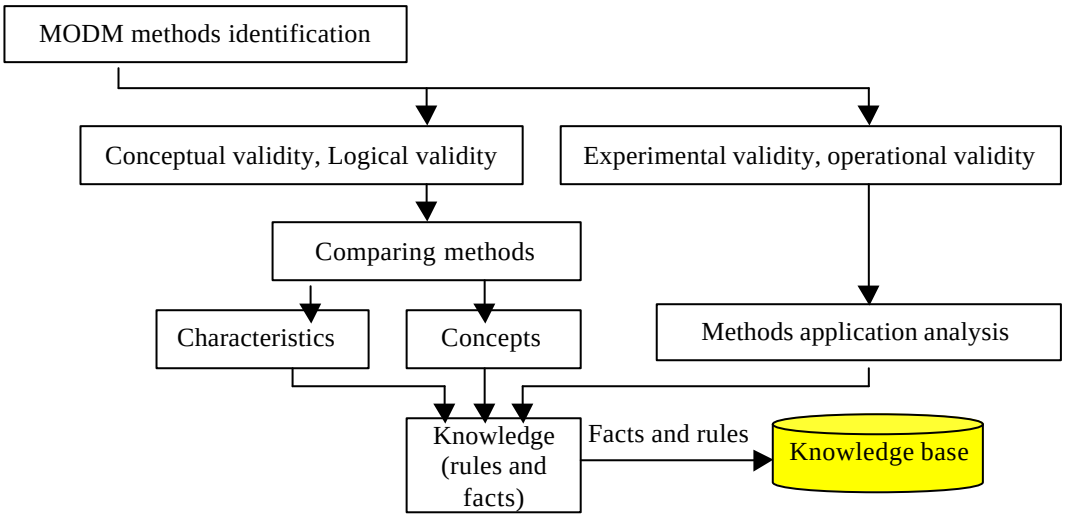

Figure 2: Knowledge acquisition process

Determining the MODM methods

The MODM model first considers a vector of decision variables $(\bar{x})$, then develops the objective functions $(f_i(\bar{x}),\ i=1,2,\ ..,\ l)$ and constraints $(g_j(\bar{x}) \le b_j,\ j=1,2,\ ..,\ m)$. The alternatives are implicit in the feasible set characterised by the constraint's set that could be infinitely many. Mathematically, an MODM problem can be represented as follows (Hwang and Masud, 1979):

$$\left.\begin{array}{l} Maximize\ \{ f_i(\bar{x}),\ i = 1,.....,l \} \\ Subject\ to: g_j(\bar{x}) \le b_j,\ j = 1,....,m \end{array}\right\}$$

Through an extensive literature review, seven well-established MODM methods have been identified and selected for this research. These methods are: Efficient Solution via Goal Programming (ESGP) (Ignizio, 1981), Interactive Multiple Objective Linear Program (IMOLP) (Quaddus and Holzman, 1986), Interactive Sequential Goal Programming (ISGP), (Masud and Hwang, 1981), Linear Goal Programming (LGP) (Ignizio, 1976), Step Method (STEM) (Benayoun et al., 1971), Steuer (Steuer, 1977) and Zionts and Wallenius (ZW) (Zionts and Wallenius, 1976). These methods are developed as independent executables, to facilitate the flexibility required of the system.

Analyzing the Characteristics of the MODM Methods

To build the knowledge base for our system, we first structured the basic knowledge and characteristics of the selected methods. Based on the works of Tecle and Duckstein (1992) and Poh (1998), the various characteristics of the MODM methods are classified into four classes as: *DMs-related, Methods-related, Problems-related and Solutions-related*. By

studying the characteristics of seven methods, four characteristics analysis models based on the seven MODM methods are produced.

The *DMs related* characteristics analysis model includes the characteristics that are related to the DM preference for selecting a method to solve a decision-making problem. Some of these characteristics include the DMs' desire to interact with the system, and the DMs' ability to provide data for a specific MODM method. The *Methods related* characteristics analysis model consists of the characteristics that are related to the solution process of MODM methods, such as whether to use a linear or goal programming technique, whether to define an ideal solution etc. The *Problems related* characteristics analysis model includes the characteristics that are dependent on the actual decision problem. For example, some MODM methods such as IMOLP and LGP require the provision of weights for each objective. ISGP and LGP need to provide the goals for each objective. The *Solutions related* characteristics analysis model consists of the characteristics that are related to the types of solution process. Table 1 shows the solutions related characteristics analysis model. Some MODM methods such as ESGP, ISGP, LGP produce only a subset of the efficient solutions, while others such as STEUER produce all efficient solutions in the neighborhood.

Table 1. Solutions related characteristics analysis model

NO	Categorization of Characteristics	ESGP (1)	IMOLP (2)	ISGP (3)	LGP (4)	STEM (5)	STEUER (6)	ZW (7)
1	Provides All Efficient Solutions	Y	N	N	N	N	Y	N
2	Provides Single Solution in a Cycle	N	Y	N	Y	Y	N	N
3	Provides a Subset of (Efficient) Solutions in a Cycle	Y	Y	Y	Y	N	Y	Y
4	Satisfactory Solution Selected by the DMs	Y	Y	Y	Y	N	Y	Y
5	Satisfactory Solution Selected by the System (Algorithm)	N	Y	N	Y	Y	N	N
6	Decreases/Increases the Value of the Objectives in a Given Solution	N	N	N	N	N	N	Y
7	Degrades/Improves the Value of the Objectives in a Given Solution	Y	N	N	N	N	N	N
8	Sacrifices/Improves the Value of Objectives in a Given Solution	N	Y	N	N	N	N	N
9	Relaxation of Objectives in Given Solutions	N	N	N	N	Y	N	N

Y---yes, N---no.

CHARACTERISTIC-METHOD MODELS

Recognition of the Characteristics of MODM Methods

The four characteristic analysis models of the MODM methods have been produced from different aspects of the MODM research. In order to ensure the consistency of knowledge in a knowledge base, the principle of assimilation is applied for combining the characteristics in each characteristic model and to produce the characteristic-method models (Ralph and Hugh, 1995). To obtain the final set of characteristics, we made the normalization by using the following operations: (1) Join reduplicated items among the four characteristics analysis

models, and (2) Remove the items that don't possess the operation validity, experimental validity or logical validity.

Classification of Characteristics of MODM Methods

To provide the appropriate guidance for the DMs possessing different levels of MODM knowledge, we capture the characteristics into two groups in order to build the question models as a front-end of the knowledge base. Two groups of characteristics are provided, namely the *novice* and *intermediate* modes.

The novice mode includes non-technical characteristics that are applied to the DMs who are totally unfamiliar with MODM. The novice mode will correspond to a set of general non-technical questions regarding the decision problem, the expected solution(s), and the DMs' preferences. From the answers obtained, suitable methods will be recommended. A total of 10 characteristics are identified for the novice mode (Table 2). The intermediate mode is designed for the DMs who are familiar with some concepts and methods of MODM, or not so familiar with the methods but have a basic knowledge of data analysis during the solution process. The technical model consists of 14 characteristics of methods. It is used to find methods corresponding to a set of inputs for the DMs using the intermediate mode.

Table 2. Characteristics and facts related for Novice mode

NO	Name	Definition	Facts
1	Interaction	more interaction with the system	Char1
2	Subset	system provides a set of solutions	Char2
3	Unique	system provides unique solution	Char3
4	S-Selection	select one satisfactory solution by system	Char4
5	D-Selection	select one satisfactory solution by yourself	Char5
6	Analyze	analyze the solutions (e.g. improving/ sacrificing the value of objectives)	Char6
7	Ideal	system defines an ideal solution	Char7
8	Weight	prepare the weight for every objective	Char8
9	Goal	prepare the goal for every objective	Char9
10	Priority	prepare the priorities for every objective	Char10

Logical Connectivity of Methods and Their Characteristics

Bui and Sivasankaran (1988) discussed 4 multiple attribute decision-making (MADM) methods for matching their 9 assertions. Poh (1998) identified the relationship between 17 MADM methods (some of them were not implemented) and their 19 characteristics. In our project, 7 MODM methods included in IMODSS are thoroughly studied and classified according to one or more of the 10 characteristics for the novice mode and 14 characteristics for the intermediate mode. Figure 3 shows the logical connectivity between the MODM methods and the 10 characteristics for the novice mode. As an example, the ISGP method is characterized by the characteristics of the 'interaction', 'subset', 'D-selection', 'ideal' and 'goal'.

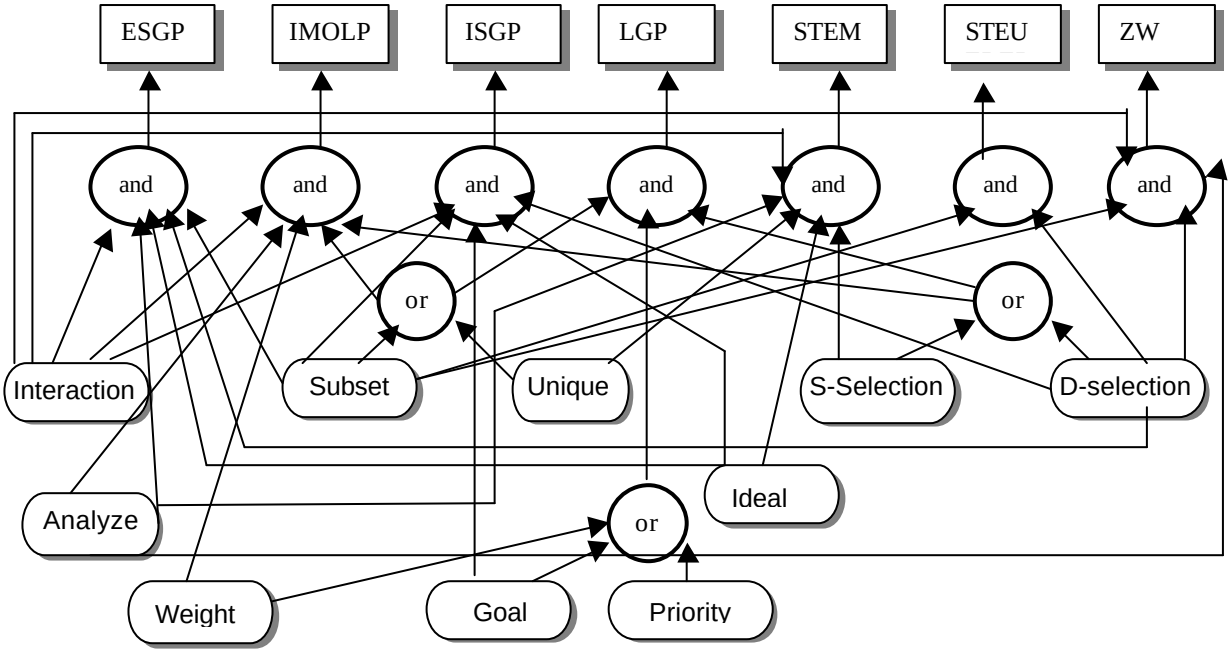

Figure 3: Logical connectivity between MODM methods and their characteristics

DESIGN OF QUESTIONS AND REPRESENTATION OF RESPONSES

Questions Design

Based on the characteristic-method connectivity, we created two groups of questions. These questions are shown to the novice or intermediate DMs through a series of dialog boxes. A group of questions for the novice DMs are shown in Table 3.

Table 3. Questions for Novice

No	Questions
Q1.	Would you like to have more interaction with the system?
Q2	Would you like the system to provide a set of solutions or a unique solution?
Q3	Would you like the system to select one satisfactory solution or would you like to select a solution?
Q4	Would you like to analyze solutions (e.g. improving/sacrificing the value of objectives)?
Q5	Would you like the system to define an ideal solution?
Q6	Have you prepared a weight for every objective?
Q7	Have you prepared a goal for every objective?
Q8	Have you prepared a priority for every objective?

Representation of Responses

When this intelligent guidance system starts up, a set of questions is displayed in a series of "Radio Group" dialog boxes. Every radio group includes one question and two response items: Top (T) and Bottom (B) and four values of weight: very important, important, general and less important. Figure 4 shows a question dialog box. The DMs can choose one of the responses and one of the degrees of importance, and then go down to the next question. They can also go back to the previous question to change their responses, or exit the question dialog box in any question to accept the default for the responses.

Question-Response-Characteristic

These responses are used to match the characteristics of one method and the weights are used to measure which method is the most appropriate if no completed method matches with one DM's preferences. The relationships between questions, responses and characteristics are shown in Table 4 for the novice mode.

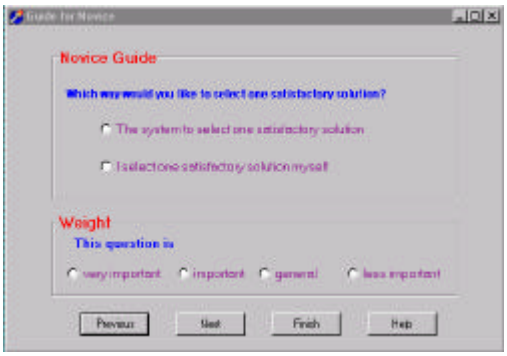

Figure. 4. A question dialog box for Novice

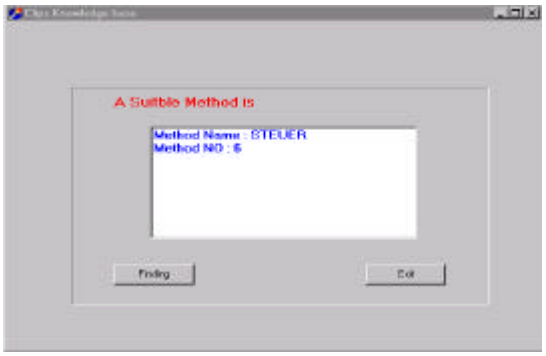

Figure. 5. A recommendation for method selection

Table 4. Question-answer-characteristic for novice mode

Question No	Answers	Characteristics' name	Characteristics' No
Q1.	T	Interactive	1
Q1	B	Not	Not
Q2	T	Subset	2
Q2	B	Unique	3
Q3	T	S-Selection	4
Q3	B	D-Selection	5
Q4	T	Analyze	6
Q4	B	Not	Not
Q5	T	Ideal	7
Q5	B	Not	Not
Q6	T	Weight	8
Q6	B	Not	Not
Q7	T	Goal	9
Q7	B	Not	Not
Q8	T	Priority	10
Q8	B	Not	Not

Method Match Process

The responses and weight marks are converted to a response vector R that consists of the characteristics the DM needs, and a weight vector W that consists of the weights of each characteristic. If a DM's completed responses match with a method, this method is recommended by this system. However, it is not often that a DM's responses exactly match the characteristics of one method. In this case, a lowest weight element W_l is obtained from weight vector W, and a characteristic that according to W_l is found and ignored**.** The knowledge base then tries to find a method which is 1-step matched with the DM's requirement. If a method is found, this method is the closest method with the DMs' request, else the above steps are repeated until a n-step matched method is found. The objective of this method is to combine the DMs' preferences and the weights for each characteristic to find a most suitable method that best satisfies the DM's requirement. This system allows the DMs to default the weight assignment. The system will provide a weight vector W automatically based on the knowledge of the degree of importance of each characteristic. Figure 5 shows a recommendation for the method selection based on the n-step matched approach.

DESIGN OF THE KNOWLEDGE BASE

Composition of MSKB

A production rules based knowledge base system has the following main characteristics: (1) the knowledge base system separates the declarative and procedural knowledge, (2) the operations of a knowledge base system are driven by facts such that a process is activated provided that certain conditions (facts) are present (Klein and Methlie, 1995).

The knowledge base system is constructed by a set of facts to define the knowledge of the methods and a set of rules that are used for finding a suitable method. These rules together with the facts form the entire knowledge base, MSKB. The MSKB is embedded in the intelligent guidance subsystem which also consists of the question subsystem, response subsystem, method-show subsystem, characteristic missing (ignore) subsystem and main-control subsystem.

Design of the Facts in MSKB

Facts are one of the basic high-level forms for representing information in a knowledge based system. Each fact represents a piece of information that has been in the current list of facts. Facts are the fundamental unit of data used by rules. The MSKB includes several groups of facts that have different functions. The basic knowledge about each MODM method and its various characteristics are described by a group of facts, namely "method". Another group of facts relates the response of each question to the facts to be asserted by the inference engine into the working memory, namely "response". Another group of facts relate each characteristic to its corresponding question, namely "characteristics-question". The next group of facts relates a question to another question to follow depending on the response, namely "question-question". It is necessary to get a set of facts to relate facts that are grouped under the same class, "characteristics-class". We also need to get a set of facts to initialize the inference process, get data and so on. The knowledge is then represented using 'def-templates' and 'def-facts'. The def-template defines a group of related fields in a pattern similar to the way in which a record is a group of related data. Definitions of three pieces of def-templates and def-facts are as follows:

(1) **method**: seven MODM methods and their various characteristics.

```
(deffacts  method1
                  (Method
                     (Number 1)
                     (Name ESGP)
                     (Char1 interaction)
                     (Char2 subset)
                     (Char5 d-selection)
                     (Char6 analyze)
                     (Char7 ideal)
                  )
)
```

(2) **response**: A set of facts relating the response of each question to the facts to be asserted by the inference engine into the working memory;

```
 (deftemplate Response
                  (field Question
                    (type INTEGER)
                    (default ?NONE)
                  )
                  (field Answer
                     (type INTEGER)
                     (default 0)
                  )
)
```

(3) **characteristics-question**: A set of facts relating each characteristic to its corresponding question;

```
(deftemplate Char-to-Quest
                  (field Char
                     (type SYMBOL)
                     (default ?NONE)
                  )
                  (field Quest_No
                     (type INTEGER)
                     (default ?NONE)
                  )
)
```

Design of Rules in the MSKB

Rules are used to represent heuristics to specify a set of actions to be performed for a given situation. We defined a set of rules which collectively work together to solve the method selection problem. The MSKB system attempts to match all the characteristics of a method to those already asserted into the working memory. If the match fails, a characteristic(s) of the last question(s) will be omitted. A method will be selected if all its characteristics (or after omission) are found in the working memory. We have also incorporated many heuristics that assist the system in the conflict resolution phase of the inference. For example, the rule to inform the user that a suitable method has been found shall have priority over other rules. Definitions of two rules are shown as follows**:**

(1) **call-question**: a rule relating to get the questions' number and its responses' number.

```
(defrule get-question
          (declare (salience 10))
          ?v1 <- (Question (Number ?num1))
          (test (neq ?num1 -1))
=>
          (retract ?v1)
          (bind ?response (quest ?num1))
          (assert (Response (Question ?num1) (Answer ?response)))
)
```

(2) **question-action:** after asking the DMs a question and getting the response, the rule checks and compares the question number, answer number and facts between facts question_answer_action and response. If the number of question and answer match one of the facts' question_answer_action, the numbers of question and fact are asserted.

```
    (defrule quest_action
         (declare (salience 20))
         ?v1 <- (Q_A_Action (Question ?num1) (Answer ?ans1) (Facts $?facts))
         ?v2 <- (Response (Question ?num1) (Answer ?ans1))
          (test (neq ?facts no))
    =>
         (retract ?v1 ?v2)
         (assert (Data (Question ?num1) (Facts ?facts)))
    )
```

All patterns must be satisfied by the facts in the fact-list for the rules to fire. A program will not start running unless there are rules whose left-hand side (LHS)'s are satisfied by the facts. The inference engine sorts the activations according to their salience. This sorting process eliminates the conflict of deciding which rule should fired next.

IMPLEMENTATION OF MSKB SYSTEM

Development Environment of MSKB System

We applied an expert system tool called CLIPS to implement the MSKB system. CLIPS is a complete environment for developing expert systems and it provides the basic elements of an expert system such as the fact-list, the knowledge-base which contains all the rules, the rule-

base, and the inference engine that controls overall execution of the rules. There are three ways to represent knowledge in CLIPS. One is by rules, which are primarily intended for heuristic knowledge based on experience. The second one is by defining functions, which are primarily intended for procedural knowledge. Object-oriented programming rules, also primarily intended for procedural knowledge, may pattern match on objects and facts. The three knowledge representation ways are all used in the MSKB system. CLIPS offers seven different modes of conflict resolution: depth, breadth, LEX, MEA, complexity, and random. We use “depth strategy” in our application. In the depth strategy, new activations are placed on the agenda after activations with higher salience, but before activations with equal or lower salience.

CLIPS is designed to be embedded within other programs. When CLIPS is used as an embedded application, the system must provide a main program to call CLIPS. The main system and other subsystem of IMODSS have been developed in the Delphi environment. CLIPS is written in C and so it cannot be called directly from Delphi. We applied a package that contains the files for encapsulating CLIPS in Delphi through the use of DLLs. We then created a new component "Tclips" in Delphi to encapsulate the Clips support files, which includes CLIPSSupport, ClipsFact, CLIPSHeader and CLIPSrules. Through this encapsulated system support, various CLIPS’ functions can be called in the Delphi based IMODSS main system.

Framework of MSKB System

The framework for the implementation of MSKB is shown in Figure 6. The guidance subsystem first questions the DMs using an elicitation technique. The responses are sent to the response subsystem and are asserted into the knowledge base system as the facts. The facts of the methods’ characteristics are asserted in the working memory by the inference engine. The knowledge base system will attempt to match the response-based characteristics of the methods to the characteristics already asserted in the working memory. A suitable method(s) will be found once all its characteristics are matched with those in the working memory. The name of method(s) then will be displayed to DMs at the end of the inference process. If a completed matched method doesn't exist, the missing characteristic approach will be executed, and an n-step matched method with the lowest match degree *n* is provided after a “Find” button is clicked.

By requesting assistance, the system runs the inference engine that automatically calls the questions programming to decide which method should be selected. The guidance system works by depending only on inputs from the DMs and determines the best MODM method that matches the needs of the specific problem. With the architecture of the intelligent front-end, IMODSS can achieve better integration of different models and allow selective and flexible use of many popular methods.

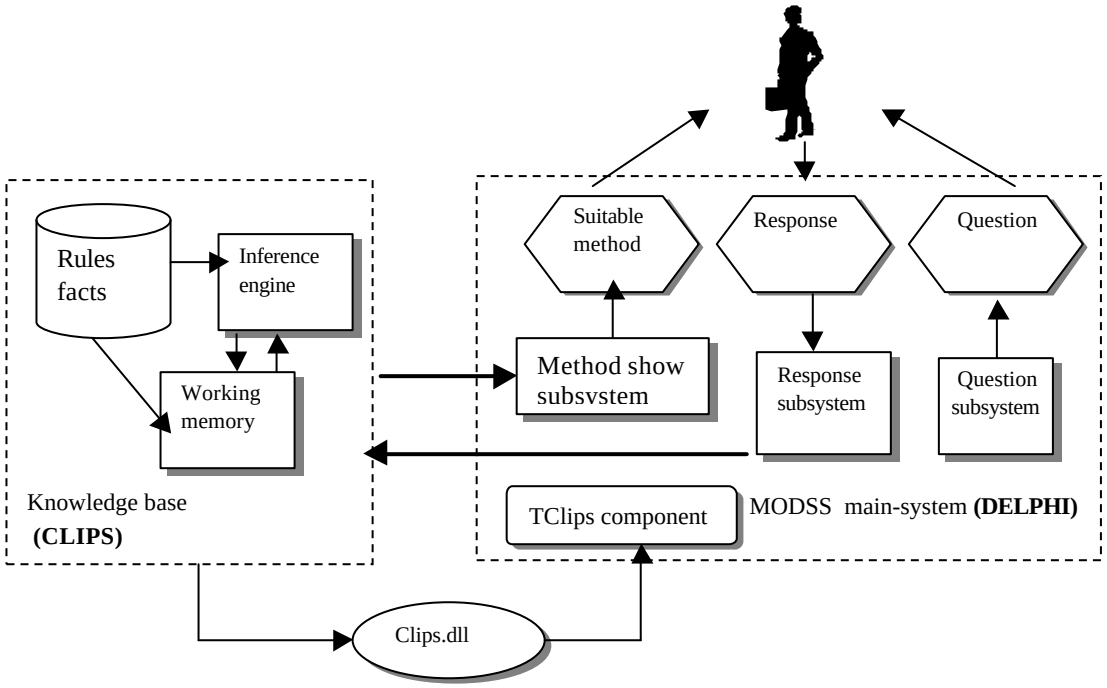

Figure 6: Design of a knowledge-based guidance subsystem for method selecting

CONCLUSIONS

A knowledge-based MODSS enhances MODSS when expert system technologies are smoothly integrated. The contributions of this research lie in the creation of a knowledge-based multiple objective decision support system, particularly, a knowledge-based intelligent guidance subsystem for selecting a suitable MODM method(s).

Since organizational decisions are primarily taken in a group environment, group decision-making (GDM) involving MODM has also generated much interest as evidenced by the literature. Our current research is to develop a group subsystem based on IMODSS to form an intelligent multiple objective group decision support system (IMOGDSS). A knowledge-based intelligent guidance subsystem can be embedded in the group system in order to provide guide during the whole group decision process. While this intelligent guidance subsystem is applied in a group environment, each member can receive guidance serially during the solution process based on his/her requirements and each member can accept a recommendation for an appropriate method selection.

REFERENCES

Benayoun, R., de Montogolfier, J., Tergny, J. and Larichev, O. (1971) Linear Programming with

Multiple Objective Functions: Step Method (STEM), *Mathematical Programming*, 1, 3, 366-375.

Bui, X. T. and Sivasankaran, T. R. (1988) An Intelligent Front End for MCDM Based Decision Support Systems, *Proceedings of the VIIIth International Conference on Multiple Criteria Decision Methods*, Manchester, England.

Hwang, C. L. and Masud, A. S. M. (1979) *Multiple Objective Decision Making - Methods and Applications: A State of the Art Survey*, Springer, New York.

Ignizio, J. P. (1976) *Goal Programming and Extensions*, Lexington Books, Massachusetts.

Ignizio, J. P. (1981) The Determination of a Subset of Efficient Solutions via Goal Programming, *Computer and Operations Research*, 8, 9-16.

Klein, M. R. and Methlie, L. B. (1995) *Knowledge-Based Decision Support Systems with Applications in Business*, John Wiley and Sons.

Lu, J., Quaddus, M. A. and Williams, R. (1999) A Framework and Prototype for Intelligent Multiple Objectives Decision Support System, *Proceedings of 4th Asia Pacific Decision Sciences Institute Conference*, Shanghai, China.

Masud, A. S. M. and Hwang, C. L. (1981) Interactive Sequential Goal Programming, *Journal of the Operational Research Society*, 32, 391-440.

Pinson, S. and Moraitis, P. (1996) An Intelligent Distributed System for Strategic Decision Making, *Group Decision and Negotiation*, 6, 77-108.

Poh, K. (1998) Knowledge-Based Guidance System for Multi-Attribute Decision Making, *Artificial Intelligence in Engineering*,12, 3, 315-326.

Poh, K. L., Quaddus, M. A. and Chin, K. L. (1995) "MOLP-PC: An Interactive Decision Support Environment for Multiple Objective Linear Optimization", in M. C. T. L. Goh (ed.) *OR Applications in Singapore*, Operational Research Society of Singapore, 25-39.

Quaddus, M. A. and Holzman, A. G. (1986) IMOLP: An Interactive Method for Multiple Objective Linear Programs, *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-16, 3, 462 -468.

Ralph, H. S., Jr. and Hugh, J. W. (1995), *Decision Support for Management*, Prentice Hall, New Jersey.

Steuer, R. E. (1977) An Interactive Multiple Objective Linear Programming Procedure, *TIMS Studies in the Management Sciences*, 6, 225-239.

Tecle, A. and Duckstein, L. (eds) (1992) *A Procedure for Selecting MCDM Techniques for Forest Resource Management*, Springer-Verlag, New York.

Zionts, S. and Wallenius, J. (1976) An Interactive Programming Method for Solving the Multiple Criteria Problem, *Management Science*, 22, 6, 652-663.

ACKNOWLEDGEMENTS

The authors gratefully thank the referees of this paper for their conscientious evaluations and constructive suggestions.

COPYRIGHT