KNOWLEDGE SHARING AND NEGOTIATION SUPPORT IN MULTIPERSON DECISION SUPPORT SYSTEMS

Matthias Jarke

May 1985

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

Working Paper Series

CRIS #94

GBA #85-37(CR)

KNOWLEDGE SHARING AND NEGOTIATION SUPPORT

IN MULTIPERSON DECISION SUPPORT SYSTEMS

Abstract

A number of DSS for supporting decisions by more than one person have been proposed. These can be categorized by spatial distance (local vs. remote), temporal distance (meeting vs. mailing), commonality of goals (cooperation vs. bargaining), and control (democratic vs. hierarchical). Existing frameworks for model management in single-user DSS seem insufficient for such systems.

This paper views multiperson DSS as a loosely coupled system of model and data bases which may be human (the DSS builders and users) or computerized. The systems components have different knowledge bases and may have different interests. Their interaction is characterized by knowledge sharing for uncertainty reduction and cooperative problem-solving, and negotiation for view integration, consensus-seeking, and compromise.

Requirements for the different types of multiperson DSS can be formalized as application-level communications protocols. Based on a literature review and recent experience with a number of multiperson DSS prototypes, artificial intelligence-based message-passing protocols are compared with database-centered approaches and model-based techniques, such as multicriteria decision making.

1.0 INTRODUCTION

Model building and application are universal features of human and computerized problem solving. A DSS needs models of the problems to be solved, of the data to be used in the problem-solving process, and of the users who are trying to solve the problems. The DSS architecture proposed by Sprague and Carlson (1982) captures these three tasks by sub-dividing a DSS in the three components of model manager, data manager, and dialog manager. Traditionally, the DSS has been perceived as a homogeneous single-user system which interacts -- at different times -- with two kinds of users: the DSS builder (typically a systems analyst with substantial expertise in computers), and the decision maker (typically with very limited computer skills). Current Model Management Systems (MMS) preserve a fairly strict distinction between these user types and provide few facilities for user or systems learning during problem-solving [1].

More importantly, these systems fail to recognize that organizational decision-making is rarely a one-person activity (Keen and Scott-Morton, 1978; Bonczek et al., 1979; Elam et al., 1980; Dolk, 1984). On the one hand, multiple decision makers may participate in a decision, in a cooperative group setting or via bargaining-type negotiations among multiple parties. These users may feel a need to communicate and negotiate not only directly but also through their DSS. On the other hand, each decision maker may want to consult multiple models, knowledge bases, and databases Frequently, these knowledge sources rely on inconsistent assumptions and different perceptions of the problem. Traditional DSS provide little support in such situations.

[1] A notable exception are the popular spreadsheet systems which, however, do not offer very sophisticated modelling capabilities.

The apparent need for integrating the ever-increasing number of micro-based DSS (Bernard, 1979; Meador et al., 1984) suggests a radically different approach to model management which takes into account the existence of multiple interacting DSS as well as multiple interacting users. This approach -- based on Hewitt's (1976, 1985) concept of "Open Systems" -- perceives a distributed DSS as a collection of loosely coupled problem solvers (human or computerized) which communicate in two different modes: a knowledge acquisition or learning mode, and a problem-solving mode. Figure 1 presents the main types of interactions for both modes, between systems, between users, and between system and user.

This paper is an attempt to address some of the general model management questions in such a multiperson DSS context. The composition of non-homogeneous mental and computerized models cannnot be based on model inputs and outputs alone, as has been suggested in single-user modelling contexts (Blanning, 1983; Sivasankaran and Jarke, 1985). Rather, model composition requires a careful analysis of the assumptions underlying each model. Moreover, human as well as computerized components of the distributed DSS may not always be accessible. Enlisting their communication and cooperation may require an elaborate negotiation process.

In summary, it appears that the design of the communications subcomponent of the distributed MMS is crucial for the success of a multiperson DSS. In the sequel, we shall first examine the communication requirements of such systems in more detail based on a taxonomy of multiperson DSS. Next, we study how three of the major DSS "parent" areas, databases, artificial intelligence, and operations research have attempted to deal with multiperson systems. Finally, we summarize our conclusions concerning a general communications framework for multiperson DSS.

2.0 A TAXONOMY OF MULTIPERSON DSS

The communications needs and opportunities of a multiperson DSS are largely dependent on the setting in which the multiperson decision takes place. In this section, we propose four dimensions that may assist in classifying multiperson DSS requirements: spatial distance among the decision makers, temporal distance among the decision-making activities by individual group members, commonality of goals among the decision makers, and type of control over the multiperson decision process. Although there is a continuum of possibilities along each dimension, they will be dichotomized here for simplicity.

Spatial distance. This dimension determines whether full face-to-face communication among decision makers is possible in addition to using the DSS. While this important feature is present in local DSS situations (Huber, 1982), a remote multiperson decision setting must compensate for its lack by providing electronic communications facilities for all aspects of the multiperson decision-making process.

Temporal distance. This dimension determines whether decisions are made by meetings at a particular point in time, or whether decision makers submit their input at different points in time. Examples of the former setting include conventional meetings but also teleconferencing, whereas the latter may be based on concepts of electronic mail, bulletin boards, and computerized conferencing (Turoff and Hiltz, 1982).

Commonality of goals. This dimension distinguishes a situation in which a group wants to solve a common problem cooperatively, from one in which (potentially hostile) parties are bargaining. Research in multiperson DSS has mostly addressed the first problem (DeSanctis and

Gallupe, 1984). Here, the main issues are knowledge sharing among multiple experts, preference aggregation, and negotiation in a friendly setting. Only recently, DSS researchers have tried to exploit the theoretical results obtained by behavioral and operations research for bargaining situations.

Control. This dimension distinguishes between situations in which the decision makers reach a decision in a democratic process, and a setting in which there is a human group leader or mediator. In a "democratic" setting, communication and coordination are achieved directly by the users (through the DSS in remote multiperson DSS). As a consequence of this increased system power, DSS design has to go great lengths towards system fairness -- otherwise the system may not be used by those who feel discriminated against. The same is also true if the multiperson DSS supports a human mediator who cannot impose decisions on he parties. On the other hand, if there is a more powerful group leader or compulsory arbitration, the multiperson DSS in final consequence is mostly a DSS for this person, and should be designed accordingly.

The four dimensions are summarized in Figure 2. With two values for each dimension, there are sixteen types of multiperson decision settings. Each of these can then be mapped on a communications design which is either based on point-to-point communications, or relies on broadcasting of messages. Thus, there are at least 32 types of multiperson DSS only a few of which have been explored in actual systems. For example, Huber's (1982) "decision room" example explores a cooperative, local, meeting-oriented boradcasting concept with a group facilitator as a (weak) leader. His Delphi example, on the other hand, is distributed both in time and space, and mostly point-to-point. The system Co-oP described in (Bui and Jarke,

1984) supports temporally and spatially remote cooperative decision-making with democratic control and mostly broadcasting messages. A spatially remote cooperative meeting setting on a point-to-point basis is implemented in many computer centers to allow consultants to track errors with remote users on-line. Jarke (1982) describes a hierarchically distributed DSS for container management in a spatially and temporally distributed setting with strict hierarchical control and point-to-point communication; here, the multiperson DSS degenerates to an implementation of problem decomposition.

Space restrictions prevent further elaboration on the different types of multiperson DSS. However, it should have become obvious that there is a rich field for further research. The remainder of this paper investigates application-level communications technologies that can be borrowed from DSS "parent" disciplines to support multiperson decision-making. We omit a discussion of message-passing at the lower levels of the communications protocol hierarchy (Tanenbaum, 1981).

3.0 DATA SHARING IN DBMS

The idea of data "sharing" was central to the initial development of mainframe database management systems (DBMS). Shared databases reduce data entry costs and provide centralized management of data integrity. Unfortunately, there are two disadvantages of current DBMS concepts for multiperson DSS.

The first disadvantage stems from the current concept of a database transaction (Gray, 1981) which is geared more towards concurrency control than towards information exchange. The underlying assumption is that each database transaction is an atomic operation on the database with no

information other transactions. Consequently, DBMS use a concept of serializability which states that the effect of the concurrent execution of a set of transactions must be equal to that of any serial execution (Bernstein and Goodman, 1982). This can be a severe disadvantage if the purpose of a user transaction is communication with another user, i.e., the read transaction of the receiver should follow the write transaction of the sender. Further, since transactions are supposedly independent, no mechanisms are provided for them to communicate with each other directly.

Secondly, most current DSS reside on microcomputers. While the need for data management in such DSS has been recognized early on, attempts to integrate microcomputer DSS databases with each other and with centralized mainframe databases are a more recent phenomenon. The commercial solutions are ad-hoc rather than based on any specific theory.

In (Jarke et al., 1984; Jelassi, 1985; Jelassi et al., 1985) we have defined an approach to this problem that integrates data staging, microcomputer database management, multiple criteria model base management, and menu-driven dialogues from a database perspective, and drives their invocation from an abstraction mechanism in the data dictionary. While this model provides a conceptually clean (and largely implemented) solution to accessing shared mainframe databases, it does not address the problem of sharing aggregated problem representations and results.

Moreover, although the data staging mechanism is defined in a way that allows for updates to the extracted database views, the issue of concurrency control for these views is unresolved since DSS transactions may take a long time. It would be unreasonable to expect that the database remains unchanged in between. Ries (1985) emphasizes the role of integrity

constraints in micro-mainframe DBMS because they may prevent unacceptable updates where traditional concurrency control methods fail. Additionally, however, there will also be a need for compensating subtransactions that selectively undo inconsistent changes without rolling back all the work done; this can be achieved through a concept of "nested" transactions (Gray, 1981).

In summary, existing DSS databases provide little support for true data sharing and information exchange, as required by most multiperson DSS. Among the more recent concepts that may improve this situation are:

1. nested transactions that allow selective compensating subtransactions if previous decisions and changes to the database prove inconsistent. Note, that in the distrbuted DSS context, inconsistency cannot be detected or prevented in advance at acceptable costs;

2. cooperating transactions that allow point-to-point communication for local coordination among database transactions. We believe that in the distributed DSS context a message sent between user transactions is preferable to attempts to formally analyze the intent of transactions.

3. data-driven model management that allows the invocation and composition of models based on information stored in the data dictionary, thus attempting to avoid discrepancies among data and model management.

4.0 KNOWLEDGE SHARING IN KBMS

Both the database and the DSS areas have recognized the need for adding artificial intelligence (AI) capabilities to their systems (Jarke and Vassiliou, 1984). Conversely, AI researchers have recognized that their systems require better interfaces to conventional very large databases and mathematical models if they are to succeed in the business world. One emerging concept is the idea of integrated knowledge base management systems (KBMS) (Brodie and Mylopoulos, 1986). Such a KBMS would

include mechanisms for: managing complex data objects; performing inferences using symbolic integrity and deduction rules; accessing large-scale databases for conventional and unformatted (e.g., image) data; and using model libraries containing different kinds of models, such as mathematical, behavioral, or physical). In other words, KBMS are intended to integrate concepts of AI, DBMS, and DSS under one conceptual umbrella.

The full implementation of such systems is probably a decade away. However, some important aspects that distinguish KBMS from existing knowledge-based systems ("expert systems") are being investigated now. Two of these aspects appear important in multiperson DSS design: the distribution of expertise, and the changing nature of business knowledge.

Firstly, large-scale KBMS as well as multiperson DSS inherit from database systems the property of being developed and used by multiple users, both in a read and in a write mode (Figure 1). There are two purposes to this simultaneous use: knowledge sharing (Erman and Lesser, 1975) and negotiation (Davis and Reid, 1983). This has some frequently overlooked consequences for the design of knowledge-based systems and model-based DSS. The concept of stable knowledge or model correctness as an idea of truth which underlies most existing DSS and expert systems is no longer acceptable. "Knowledge" bases and "correct" models do not represent truth but the beliefs of the designers or experts, which in turn are based on their subjective assumptions and goals [2]. In a multiperson context, inconsistencies are not just a consequence of design errors which lead to annoying logical contradictions but serve as a fruitful starting point for problem-solving or compromise.

[2] Henderson (1985) points out the relevance of this observation to the use of DSS by senior executives. Here, we are focusing on its importance for distributed model management.

Carl Hewitt (1976, 1985) was among the first who pointed out the impossibility of a consistent logical theory for multi-actor problem-solving in truly distributed or decentralized decision settings. His "open systems" approach explores theoretical foundations for systems that would not even know where to ask for necessary knowledge at the beginning of a problem-solving process. Rather, the system adds new knowledge sources as the need arises. For example, a DSS following this architecture would be alerted to the existence of a particular external model or data base only during decision-making, and then try to establish a mutually understood language and knowledge exchange mechanism with that external source. A few DSS provide such "data staging" support (Sprague and Carlson, 1982; Jarke et al., 1984; Jelassi, 1985) but require substantial human intervention, e.g., in reformatting the incoming data from previously unknown sources.

Open systems grow, and their components interact, by negotiation. Davis and Reid (1983) propose a so-called "contract net" approach to use negotiations for distributed problem solving. Whenever a local problem solver cannot solve a subproblem, it broadcasts a request for proposal, describing the task at hand. Qualified other subsystems will then submit proposals, based on their current status (e.g., workload) and general capabilities. The original problem solver will then negotiate a contract with one of the bidders; among other things, the negotiations will have to make sure that the bidder is really qualified for handling the task.

As an example for extending the contract negotiation approach to DSS, consider the following model management scenario. A microcomputer-based DSS requests the solution of a fairly large linear program. Four other DSS model bases bid on this: a mixed-integer programming package, two linear

programming systems, and a network optimizer, each of them providing a cost estimate based on the problem size. The mixed-integer system is excluded because of a very high cost. First, negotiations are started with the network package; however, detailed problem analysis during negotiations shows that the problem does not have a network structure. Note, that this involves discussing the assumptions underlying the models, rather than just a simple input/output analysis. Therefore, the negotiations are abandoned. But now the machine where the fastest of the LP packages resides, is very busy and submits an increased cost estimate when approached for negotiations. Therefore, the bid by the other LP package is selected and result delivery conventions are established.

A second shortcoming of existing knowledge-based systems is that their knowledge bases are domain-specific and very stable. Initial knowledge bases can be established a priori and evolve slowly as the system acquires new knowledge. In business situations as mirrored in DSS, knowledge is often not stable at all; there are examples (e.g., stock selection) where knowledge is only useful if applied immediately, i.e., before becoming known to everybody. Moreover, multiperson DSS will be applied to varying problem contexts where little a priori knowledge is available. The multiperson DSS model manager should therefore include a machine learning component which acquires knowledge fast and based on just a few examples and existing rules (Michie, 1982; Winston, 1984).

Again, the distributed model manager must trace the source of the "learned" concepts, i.e., the decision maker or models on whose judgments the definition of the concept depends. How does the system get these assumptions? Belief maintenance systems in AI (Doyle, 1979) have proposed to ask the user to justify his decisions, and to use the justifications

(rather than an "objective" logical theory) to establish dependencies that trace the consequences of changes in assumptions. In Dhar and Jarke (1985), a method is proposed not only to record these justifications during a prototyping process but also to surface more general assumptions underlying them, and to apply those assumptions in analogy-based reasoning (Winston, 1979).

To summarize this section, AI has developed some promising concepts that could serve as a basis for model management in distributed multiperson DSS. However, neither the negotiation nor the assumption surfacing and learning concepts presented here have been fully developed or implemented for a DSS context. In particular, existing concepts do not take into account the support nature of DSS, i.e., the need to interact with multiple users as well as with multiple subsystems.

5.0 MCDM METHODS AS PROTOCOLS FOR COMMUNICATION AND NEGOTIATION

Among the operational research methods, game theory (Owen, 1982) and multiple criteria decision making (MCDM) methods were among the first to consider a multiuser context. Despite the impossibility to define axiomatically "fair" group solutions without dictatorship (Arrow, 1963), MCDM methods have a number of advantages for the multiperson DSS context (Bui and Jarke, 1984). By their very nature, they integrate multiple views of a problem, using qualitative as well as quantitative criteria. Many of the methods are interactive, allowing for easy revisions of individual or group problem representations and opinions. MCDM methods can be used in a message-passing (point-to-point) as well as in a database-centered (broadcasting-oriented) implementation, and they support democratic as well as hierarchical multiperson decision modes. By adding database

capabilities to multicriteria-based DSS, both knowledge sharing and negotiation can be supported. Our own implementation efforts have therefore focused on this area first.

A number of multiperson methods have been proposed in the MCDM literature (Bereanu, 1976; Dalkey, 1976; Keeney and Kirkwood, 1975). Some of these methods --especially if applied in reality-- were also combined with other models. Tell (1977) employs a Delphi-like method for capturing the preferences of individuals, and factor analysis for limiting the number of criteria. Kirkwood (1977) integrates MCDM with an analysis of uncertainty about decision outcomes to support multiperson analysis of public sector situations; see Goncalves (1985) for an overview of similar methods. The present author applied MCDM-based DSS in several multiplayer decision situations using single-user DSS, to assess the cost-effectiveness of large-scale public-sector information systems in banking (Jarke et al., 1981) and infection control (Mildner et al., 1984). However, all of these methods -- if computerized at all -- were implemented in a single-user albeit multiplayer mode.

In the recent past, we have been involved in a number of projects that attempt to design and build multiple user DSS for multiple criteria multiperson decision making. Co-oP (Bui and Jarke, 1984; Bui, 1985) is a DSS for cooperative group decision making implemented on a network of personal computers. There is one PC for each player and a file server used as a message-switching center (Figure 3).

Since the decision setting is assumed to be democratic, remote, and cooperative, the Co-oP design offers a wide range of communications facilities with fairly loose communications protocols, ranging from

informal electronic mail, to structured group communication tools (NGT, Delphi), to extended MCDM tools for preference aggregation and information exchange. The initial prototype of the system (Bui and Jarke, 1984) only supported a group version of one particular MCDM method. The full system (Bui, 1985) will include a larger set of models together with a rule-based system for model selection. Some behavioral, process-based tools are also being added.

If the multiperson decision situation becomes less friendly, there is a need for access control to private data and problem representations, as well as for strong tools for negotiations support. In many cases, the DSS model base will not be able to fully handle negotiation situations. Instead, the collection of human and computerized problem solvers will include a human mediator and a supporting DSS component. The mediator will help the parties (often called "players") establish a joint problem representation and then to evolve it -- through consensus-seeking and compromise -- towards a representation in which there is a mutually acceptable solution.

This concept is being implemented in MEDIATOR, a multicriteria-based micro-mainframe DSS for negotiation support (Jarke et al., 1985). The system is intended to support negotiation between the marketing and engineering departments of a European car manufacturer (Giordano et al., 1985). MEDIATOR uses a database-centered approach, i.e., most communication is achieved through manipulating database structures, similar to the "blackboard" concept in AI (Erman and Lesser, 1975). Negotiations proceed in three stages.

First, each player establishes an individual representation of the problem at hand, using publicly accessible as well as private data and DSS tools, to feed an interactive MCDM method (Jacquet-Lagreze and Siskos, 1982) that establishes the individual preference structure. This step is similar in spirit to the idea of the Nominal Group Technique (Huber, 1982) which also requires each player to come up with individual ideas first, before the discussion starts.

In the second phase, called view integration, the human mediator is supported in achieving a joint problem representation in the three steps of: database selection, alternative definition, and criteria definition. Players transfer their individual definitions of data sources, alternatives, criteria, decision matrices, and utility functions to the common database. Each player occupies a private section of that database which can be only accessed by himself and by the mediator (Figure 4). The mediator will then start the process of integrating these personal problem problem representations into the group joint problem representation. Relational operations, enhanced by redefinitions of terms, can efficiently support the view integration step.

Once this is accomplished, the joint problem representation is stored in the publicly accessible area of the common database. From then on, the "official" negotiation will only work with the joint representation. The players are free to continue using their local representation and other decision support tools for personal deliberations.

Upon successful completion of the view integration phase, the third phase, called negotiation, proceeds by consensus seeking through exchange of information and, where consensus is incomplete, by compromise. The

negotiation problem is shown --graphically or as relational data in matrix form-- in three spaces as a mapping from control space to goal space (and through marginal utility functions) to utility space (Shakun, 1985). Within each of these spaces the negotiation process is characterized by adaptive change that redefines feasible and target sets in seeking a solution.

MEDIATOR allows the human mediator to perform what-if analyses of possible suggestions for problem redefinition. Before, e.g., suggesting that players should lower their utility threshold, the mediator must make certain that this will make additional alternatives available for discussion. Otherwise, the players will feel that they made a concession for nothing and the climate of the negotiation may deteriorate.

In the control space, the relational query language offers the option of including or excluding sets of alternatives from consideration, by restricting the feasibility of certain attribute or criterion values. The human mediator can apply such queries to focus the discussion temporarily on a smaller number of alternatives, or to increase the set of feasible solutions by searching databases for decision alternatives players did not think of originally.

Changes in the goal space involve a redefinition of the criteria set: would dropping a criterion change the ranking? is the ranking by a particular criterion inconsistent with the overall utility ranking of alternatives? To answer such questions, certain display techniques for relational data are employed, most prominently alternative ranking.

The idea of ranking alternatives is also used to answer what-if questions in the utility space. Since player utilities are simply additional attributes of the decision matrix, they can be used as sorting criteria. For more than two players, it may also make sense to display the aggregated utilities of coalitions. Another representation at the utility level are overlayed marginal utility curves. What-if questions allow the mediator to vary the weights and forms of each player's utility curves tentatively, to prevent having the players agree to useless concessions.

In summary, MCDM methods can serve useful purposes as formal tools for preference surfacing, preference aggregation, negotiation, and mediation, both in friendly and in noncooperative decision situations. The close relationship of MCDM decision matrices to relational data representations (Jacquet-Lagreze and Shakun, 1984; Jelassi et al., 1985) facilitates a database-centered implementation of such concepts. However, as illustrated by the comparison between the Co-oP and MEDIATOR designs, the decision setting will have a strong influence on the question how exactly to define MCDM-based communication among multiple DSS and multiple users.

6.0 CONCLUSION

Multiperson DSS can be viewed as a community of human and computerized problem solvers with different knowledge bases and interests. Model management in such systems requires a strong communications component within the system as well as with the users. Communications design must support two systems capabilities. Knowledge sharing reduces uncertainty and integrates distributed mental and computerized models into coherent solution strategies. User coalitions and subsystems may also have different interests, problem perceptions, and goals. The communications

protocol must provide negotiation support for integrating problem views, consensus-seeking, and compromise. Negotiation support should be offered both from the viewpoint of the individual problem solver and from the viewpoint of the group as a whole (as represented by a common theory or goal, group leader, or mediator).

Finally, knowledge sharing and negotiation both require the capability to surface and question assumptions made by model builders. Assumption management in multiperson DSS will facilitate the task of building "executive support systems" (Henderson, 1985) charged with -- the surfacing and questioning of assumptions. Assumption managaement also helps obviate the traditional notion of model "correctness" and emphasizes the subjective and volatile nature of modelling: the meaning of a model is not defined externally but determined by what its designer meant!

We have not addressed organizational implementation strategies for multiperson DSS; Peter Keen (1985) points out the crucial importance of an efficient communications infrastructure on which multiperson DSS can be piggybacked. We did, however, show that all the "parent areas" of DSS research must (and can) contribute to the design of multiperson DSS. DBMS data sharing, AI concepts for knowledge representation, belief maintenance, and negotiation, and OR methods for multicriteria decisions all have to be integrated to support multiperson decisions efficiently. Moreover, process-oriented behavioral tools must also be provided in the user interface manager (interpreted here as the user-systems communication subsystem). As Huber and McDaniel (1985) point out, such DSS may then in turn influence the design of the organizations that use them. In particular, they may reduce the purported negative influence of single-user DSS on managerial communications (Sanders et al., 1985).

REFERENCES

1. Arrow, K.J., 1963. Social Choice and Individual Values, 2nd ed., Wiley, New York.

2. Bereanu, B., 1976. Large group decision making with multiple criteria, in H.Thiriez, S.Zionts (eds.), Multiple Criteria Decision Making, Springer-Verlag, pp. 87-101.

3. Bernard, D., 1979. Management issues in cooperative computing, ACM Computing Surveys 11, 1, pp. 3-17.

4. Bernstein, P.A., and Goodman, N., 1981. Concurrency control in distributed database systems, ACM Computing Surveys 13, 2, pp.185-221.

5. Blanning, R.W., 1983. TQL: a model query language fbased on the domain relational calculus, Proceedings IEEE Workshop on Languages for Automation, Chicago, pp. 141-146.

6. Bonczek, R.H., Holsapple, C.W., Whinston, A.B., 1979. Computer-based support for organizational decision making, Decision Sciences 10, pp. 268-291.

7. Brodie, M.L., and Mylopoulos, J., eds., 1986. On Knowledge Base Management Systems, Springer-Verlag, forthcoming.

8. Bui, X.T., 1985. DSS for Cooperative Group Decision Making, Ph.D. Dissertation, Department of Computer Applications and Information Systems, New York University, forthcoming.

9. Bui, X.T., and Jarke, M., 1984. A DSS for cooperative multiple criteria group decision making, Proceedings 5th International Conference on Information Systems, Tucson, Az, pp. 101-113.

10. Dalkey, N.C., 1976. Group decision analysis, in M.Zeleny (ed.), Multiple Criteria Decision Making Kyoto 1975, Springer-Verlag, pp. 45-74.

11. Davis, R., and Smith, R.G., 1983. Negotiation as a metaphor for distributed problem solving, Artificial Intelligence 20, pp. 63-109.

12. DeSanctis, G., and Gallupe, B., 1984. Information systems support for group decision making, MISRC Working Paper Series MISRC-WP-85-10, University of Minnesota.

13. Dhar, V., and Jarke, M., 1985. Learning from prototypes, submitted for publication.

14. Dolk, D.R., 1984. Model management in organizations, presented at ORSA/TIMS Conference, San Francisco.

15. Doyle, Jon., 1978. A Truth Maintenance System, AI Laboratory Memo 521, Massachusetts Institute of Technology, Cambridge, Mass.

16. EDP Analyzer, 1984. Get ready to offer group services, EDP Analyzer 22, 10.

17. Elam, J.J., Henderson, J.C., and Miller, L.W., 1980. Model management systems: an approach to decision support in complex organizations, Proceedings First International Conference on Information Systems.

18. Erman, L.D., and Lesser, V.R., 1975. A multi-level organization for problem solving using many diverse, cooperating sources of knowledge, Proceedings Fourth International Joint Conference on Artificial Intelligence, pp. 483-490.

19. Giordano, J.L., Jacquet-Lagreze, E., and Shakun, M.F., 1985. Un SIAD pour la conception de produits nouveaux: aspects multicriteres et multi-acteurs, Note de Recherche, Universite de Paris-Dauphine, Paris.

20. Goncalves, A.S., 1985. Group decision methodology and group decision support systems, DSS-85 Transactions, San Francisco, Ca., pp. 135-142.

21. Gray, J., 1981. The transaction concept -- virtues and limitations, Proceedings 7th Very Large Data Base Comference, Cannes, pp. 144-154.

22. Henderson, J.C., 1985. A methodology for identifying strategic opportunities for DSS, in (Jarke, 1985).

23. Hewitt, C., 1976. Viewing control structures as patterns of passing messages, Artificial Intelligence 8, pp. 323-364.

24. Hewitt, C., 1985. Implications of open systems, in (Jarke, 1985).

25. Huber, G.P., 1982. Group decision support systems as aids in the use of structured group management techniques, DSS-82 Transactions, San Francisco, pp. 96-108.

26. Huber, G.P., 1984. Issues in the design of group decision support systems, MIS Quarterly, 8, 3, pp. 195-204.

27. Huber, G.P., and McDaniel, R.R., 1985. Integrating organizational information processing and organizational designs, in (Jarke, 1985).

28. Jacquet-Lagreze, E., and Shakun, M.F., 1984. Decision support systems for semi-structured buying decisions, European Journal of Operations Research 16, 1, pp. 48-58.

29. Jacquet-Lagreze, E., and Siskos, J., 1982. Assessing a set of additive utility functions for multicriteria decision making: the UTA method, European Journal of Operational Research 10, 2, pp. 151-164.

30. Jarke, M., 1982. Developing decision support systems: a container management example, Policy Analysis and Information Systems, 6, 4, pp. 351-372.

31. Jarke, M., ed., 1985. Managers, Micros, and Mainframes: Proceedings NYU Symposium on Integrating Systems for End Users, New York University, New York.

32. Jarke, M., Jelassi, M.T., and Shakun, M.F., 1985. MEDIATOR: Towards a negotiation support system, forthcoming in M.F.Shakun, Evolutionary Systems Design: Policy Making under Complexity, San Francisco: Holden Day.

33. Jarke, M., Jelassi, M.T., and Stohr, E.A., 1984. A data-driven user interface generator for a generalized multiple criteria decision support system, Proc. IEEE Workshop on Languages for Automation, New-Orleans 1984, pp. 127-133.

34. Jarke, M., Puller, H., and Thornton, P., 1981. Kosten-Nutzwertanalyse Zweite Automationsphase Deutsche Bundesbank, unpublished report, Frankfurt.

35. Jarke, M., and Vassiliou, Y., 1984. Coupling expert systems with database management systems, in W.Reitman (ed.), Artificial Intelligence Applications for Business, Ablex, Norwood, NJ, pp. 65-85.

36. Jelassi, M.T., 1985. An Extended Relational Database for Generalized Multiple Criteria Decision Support Systems, Ph.D. Dissertation, Department of Computer Applications and Information Systems, New York University, May 1985.

37. Jelassi, M.T., Jarke, M., and Stohr, E.A., 1985. Designing a generalized multiple criteria decision support system, Journal of MIS 1, 4, pp. 24-43.

38. Keen, P.G.W., 1985. Highways and traffic: building the telecommunications infrastructure, in (Jarke, 1985).

39. Keen, P.G.W., and Scott-Morton, M.S., 1978. Decision Support Systems: An Organizational Perspective, Addison-Wesley, Reading, Mass.

40. Keeney, R.L., and Kirkwood, C.W., 1975. Group decision making using cardinal social welfare functions, Management Science 22, pp. 430-437.

41. Kirkwood, C.W., 1977. Social decision analysis using multiattribute utility theory, in S.Zionts (ed.), Multiple Criteria Problem Solving, Springer-Verlag, pp. 335-344.

42. Meador, C., Keen, P.G.W., and Guyote, M., 1984. Personal computers and distributed decision support, Computerworld, April.

43. Michie, D., 1982. The state of the art in machine learning, in D.Michie (ed.), Introductory Readings in Expert Systems, Gordon and Breach, UK.

44. Mildner, R., Jarke, M., and Ohgke, H., 1984. Infektionshygiene im Krankenhaus, Medizin Mensch Gesellschaft 9, 1, pp. 38-47.

45. Miller, L.W., and Katz, N., 1983. Model management systems to support policy analysis, Decision Sciences Technical Report 82-11-01, Wharton School, University of Pennsylvania.

46. Owen, J., 1982. Game Theory, 2nd ed., Academic Press, New York.

47. Ries, D.R., 1985. Distributed databases and distributed processing between personal computers and mainframes, in Jarke, 1985).

48. Sanders, G.L., Courtney, J.F., Loy, S.L., 1984. The impact of DSS on organizational communications, Information & Management 7, 2, pp. 141-148.

49. Shakun, M.F., 1985. Decision support systems for negotiations, forthcoming in M.F.Shakun, Evolutionary Systems Design: Policy Making under Complexity, Holden Day, San Francisco.

50. Sivasankaran, T.R., and Jarke, M., 1985. Logic-based formula management in an Actuarial Consulting System, Decision Support Systems 1, 3.

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

52. Stohr, E.A., 1982. DSS for cooperative decision-making, NYU Working Paper Series CRIS #19, GBA 81-27.

53. Tanenbaum, A., 1981. Network protocols, ACM Computing Surveys 13, 4, pp. 453-489.

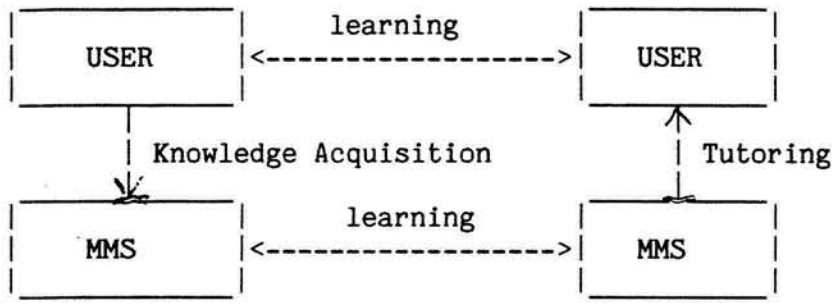
54. Tell, B., 1977. An approach to solving multi-person multiple-criteria decision-making problems, in S.Zionts (ed.), Multiple Criteria Problem Solving, Springer-Verlag, pp. 482-493.

55. Thiriez, H., 1976. Multi-person multi-criteria decision-making: a sample approach, in H.Thiriez, S.Zionts (eds.), Multiple Criteria Decision Making, Springer-Verlag, pp. 103-117.

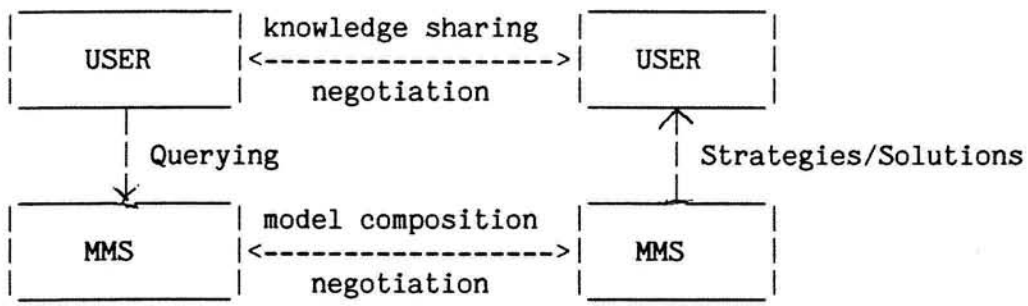
56. Turoff, M., and Hiltz, S.R., 1982. Computer support for group versus individual decisions, IEEE Transactions on Communications COM-30, 1, pp. 82-90.

57. Winston, P.H., 1979. Learning and reasoning by analogy, Communications of the ACM 23, 12, pp. 689-703.

58. Winston, P.H., 1984. Artificial Intelligence, 2nd ed., Addison-Wesley, Reading, Mass.



(a) Knowledge Acquisition Mode



(b) Problem-Solving Mode

FIGURE 1: Multiperson Model Management System (MMS) Usage Modes

spatial distance
- local
- remote

temporal distance
- meeting
- mailing

commonality of goals
- cooperation/group
- bargaining/negotiations

control
- democratic/system coordination
- hierarchical/human coordination

FIGURE 2: Taxonomy of Multiperson DSS Decision Settings

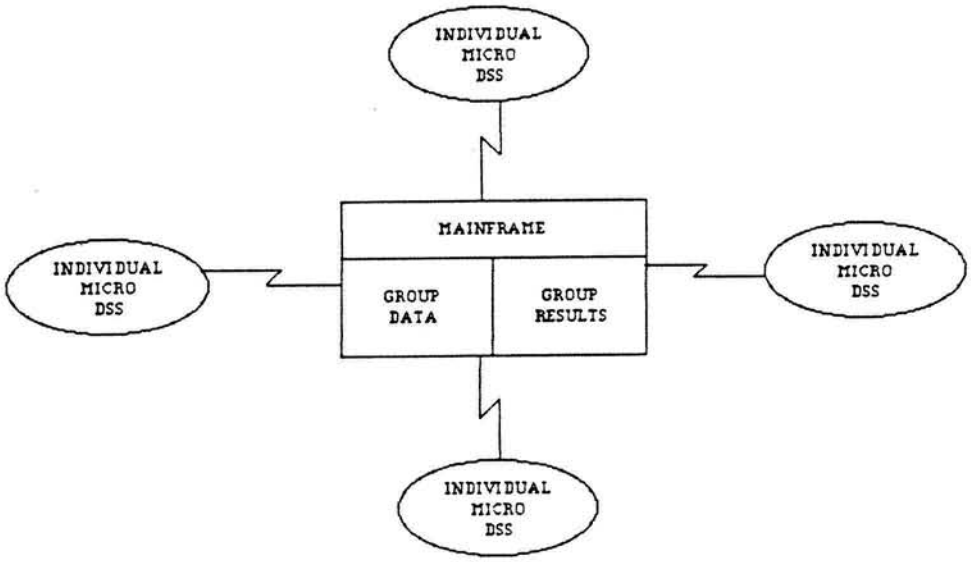


FIGURE 3: Co-oP Design -- Combining Individual and Group DSS

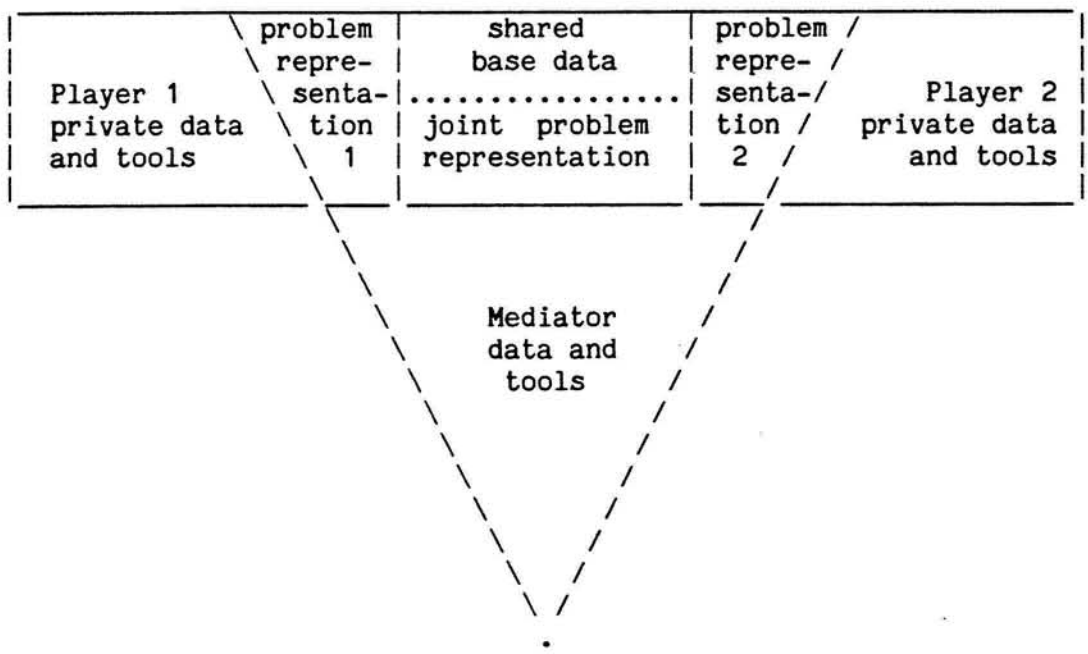


FIGURE 4: MEDIATOR Design -- Communication through Data Sharing