An integrated environment ensures better relationships between different DSS decisions.

Integrating Decision Support Systems in Organizations:

Use of Information Thechnology

Rajeev Kaula

Industrial Management & Data Systems, Vol. 94 No. 4, 1994, pp. 8-14

Introduction

Decision support systems (DSS) have contributed considerably towards the utilization of information processing systems in organizations. DSS concepts and structures have evolved from the initial stage of supporting individual decision making in the form of individual DSS (IDSS)[1,2], to a much more complex stage of supporting groups of individuals in several tasks including, but not limited to, decision making in the form of group DSS (GDSS)[3-6]. Increasingly, attempts are being made to broaden the level of decision-making support to encompass individuals, groups, divisions, departments and even entire organizations. Such broad level decision support systems have been labelled variably under the name of organization DSS (ODSS)[7-18]. Each type of DSS (IDSS, GDSS or ODSS), in spite of having a common conceptual base, has its own distinct structure and capability. With the relative maturing of each DSS conceptualization, there arises a need to integrate their usage into a single integrated framework.

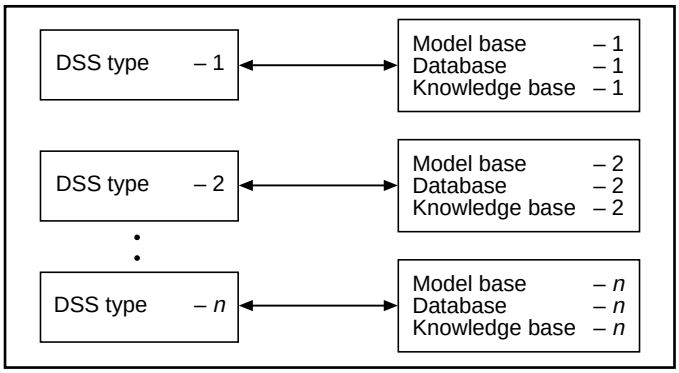
Traditionally each type of DSS is viewed and developed separately from the other. As the decision support provided by IDSS, GDSS or ODSS varies, it facilitates having a separate development approach. However, being part of the same organization, the decision support provided by these different DSS is also related one with another. For instance, in a department it is possible to have many IDSS. Each IDSS supports an individual's decision making. However, being part of the same department, these individual decisions have to conform with the department goals and objectives. This makes the separate decisions related (to some degree) to one another, in the cause of achieving departmental goals and objectives. Similarly the decision support provided by GDSS and ODSS within an organization also relates one with another.

Further, owing to the interdependent and interrelated nature of an organization[19], the decision support structures used by different DSS across the organization are not entirely unique. It is possible that many overlapping criteria and models may exist in different DSS. For instance, there could be a GDSS for developing a marketing plan, and an IDSS for a salesperson to develop a marketing strategy to achieve a plan. Although both the DSS are distinct, similar criteria and models may appear in their decision structures.

Increasingly, the existing DSS development environment (shown in Figure 1) has started resembling the file management environment associated with transaction processing systems[20]. In Figure 1, each DSS type (ranging from 1 to *n*) can be an IDSS, GDSS, or ODSS. Each DSS type has a separate set of model base, data-base, and knowledge base, making it difficult for another DSS type to share and access them. As the decisions and the associated structures supported by the different DSS are related, it is necessary to streamline the development of DSS from an organizational perspective. Such a perspective involves developing an integrated environment in which the overlapping criteria and models can be shared; and the different DSS are better able to relate with one another.

Traditionally DSS development is represented by the three-level DSS framework[1,21] consisting of specific DSS, DSS-Generator, and DSS-Tools levels. This traditional DSS framework allows for the integration of many DSS through the DSS-Generator. DSS-Generator provides a set of general technical capabilities that are adapted to individual system needs in the form of specific DSS. However, DSS-Generators have a technical connotation. For instance, if a specific DSS database requires certain data, the DSS-Generator would provide mechanisms to retrieve the data. The emphasis on integration is not mandatory, as it is also possible to develop specific DSS structures from DSS-Tools directly,

Figure 1. *The DSS Development Environment*



ignoring the integration capability provided by the DSS-Generator.

In order to resolve this issue, a framework is required that, apart from providing an integrated development environment, also allows for the different DSS to preserve their uniqueness and distinct characteristics. Having an integrated development environment will also stabilize the development and maintenance of DSS activity in the organization.

Due to the similarity of the existing DSS environment (as shown in Figure 1) with the file management environment of transaction-processing systems, one approach to achieve DSS integration is to adapt a conceptual framework that enables system integration in transaction processing systems to the existing DSS environment. In this context, the three-level ANSI/X3/SPARC database framework[22,23] is considered as a basis for developing the integrated DSS environment. The assumption being that as the three-level database framework allows for the separate development of individual systems in a uniform set-up, it may be possible to emulate this characteristic for the existing DSS environment.

This article explores the extent of integration possible among the different DSS, and provides a framework for integrating them using the ANSI/X3/SPARC three-level database framework. The article proposes a knowledge-based approach to implement the outlined DSS framework.

Problems in Existing DSS Environment towards Integration

DSS primarily rely on their model base, database and knowledge base components to provide decision support. Any attempt at integrating the DSS development environment has to consider the consolidation of the model base, database and knowledge base components. Due to the related nature of decision support provided by DSS in organizations, the existing approach to their separate development leads to the following problems:

(1) Redundancy in criteria and models usage.

(2) Separate and isolated criteria and models.

(3) Dependency between DSS and its structure components.

The problems overlap and are not mutually exclusive. The problems are discussed below.

Redundancy in Criteria and Models Usage

Redundancy exists owing to the duplication of similar decision criteria in the form of models, data, and knowledge in various DSS. Although such duplication wastes storage space, that is not the most serious consequence of this problem. The more serious consequence of this duplication is the "integrity" of a decision criteria.

A collection of overlapping decision criteria in different DSS would have integrity if they are logically consistent. This means, in particular, that the duplicated criteria agree with one other. For instance, if a marketing plan changes, then all DSS using the marketing plan decision criteria need to be updated. If all DSS do not make compatible changes, the decision support provided may be logically inconsistent.

Separate and Isolated Criteria and Models

As each DSS is developed separately, the decision criteria in the form of model, data and knowledge structures utilized by them are separated and isolated. This implies that each DSS interpretation of its decision criteria is separated from the other, although the decisions supported may be interrelated. For instance, a single marketing plan may be utilized by decision makers, first, in marketing to prepare the sales strategy, and second, in finance to merge with other plans to estimate cost/resources required. Each decision-support structure is interpreting the plan in different ways. Marketing might view the plan as a competitive strategy, while finance may view the plan from the cost/resources perspective. So, for developing the competitive strategy, the criteria and models place less emphasis on the cost aspects. Similarly for developing the cost/resource requirements, the criteria and models place less emphasis on different competing strategies. These different interpretations are valid, as long as the marketing plan is not changed. Any change in the marketing plan would require compatible changes in different DSS usage of the plan. This will become a problem if there are many interpretations of the same plan, resulting in a kind of chain effect of changes to be made.

Dependency between DSS and Its Structure Components

There is a dependency between the decision supported by a DSS and its structure components. This includes the

storage structures used to store the decision criteria in the form of models, knowledge and data. Each DSS decision criterion is developed from the perspectives of the decision maker. This dependency characteristic is intrinsic to DSS development. However, the dependency has to be weakened, if there is going to be proper sharing of a decision criteria. Strong dependency ties decision structures to a specific interpretation or context, making it difficult to share.

Framework for Integrating DSS Structures

The framework for integrating different DSS is based on the three-level ANSI/X3/SPARC database framework. A brief review of the database framework follows.

Three-level Database Framework Review

The ANSI/X3/SPARC three-level database design framework[22,23], as shown in Figure 2, organizes database development in three levels – external, conceptual and internal.

The external level is concerned with individual user or application data requirements, represented by various external schemata or subschemata. The conceptual level is concerned with the entire organization's logical data requirements, represented by the conceptual schema. The internal level is concerned with the entire organization's physical storage of data, represented by the internal schema.

The external level, and the conceptual level describe the logical structure of the database, while the internal level describes the physical structure of the database. The conceptual schema utilizes the concept of data modelling and abstraction [24] to achieve the logical structuring of data. The conceptual level also provides a link (bridge) between the users' level with its related external views, and the physical storage level with its related internal view.

Three-level Integrated DSS Framework

The three-level database framework can be adapted to the DSS environment in a compatible three-level integrated DSS framework as shown in Figure 3. The framework focuses on the representation of the three basic structural elements in DSS – model, data and knowledge. The framework has three levels represented by the external view, the conceptual view and the internal view. Each view is now discussed.

Conceptual View

The conceptual view will have three sets of interacting schemata representing the model base, the knowledge base, and the database respectively (as shown in Figure 4). The model schema is the logical specification of the various models needed by various decision makers through their specific DSS structures. The knowledge schema is the logical specification of knowledge needed to support specific DSS structures. The database schema is the logical specification of data that will be utilized by the model schema and the knowledge schema. Logically, the conceptual view is an aggregation of external views of different specific DSS. By having one repository of all models, knowledge and data it will be possible to reduce the redundancies and allow sharing. The structure of the three schemata is now described.

Model Schema

The model schema is the logical repository of all models in the form of inter-related modules called an MB-Module

Figure 2. *The ANSI/X 3/SPARC Three-level Database Design Framework*

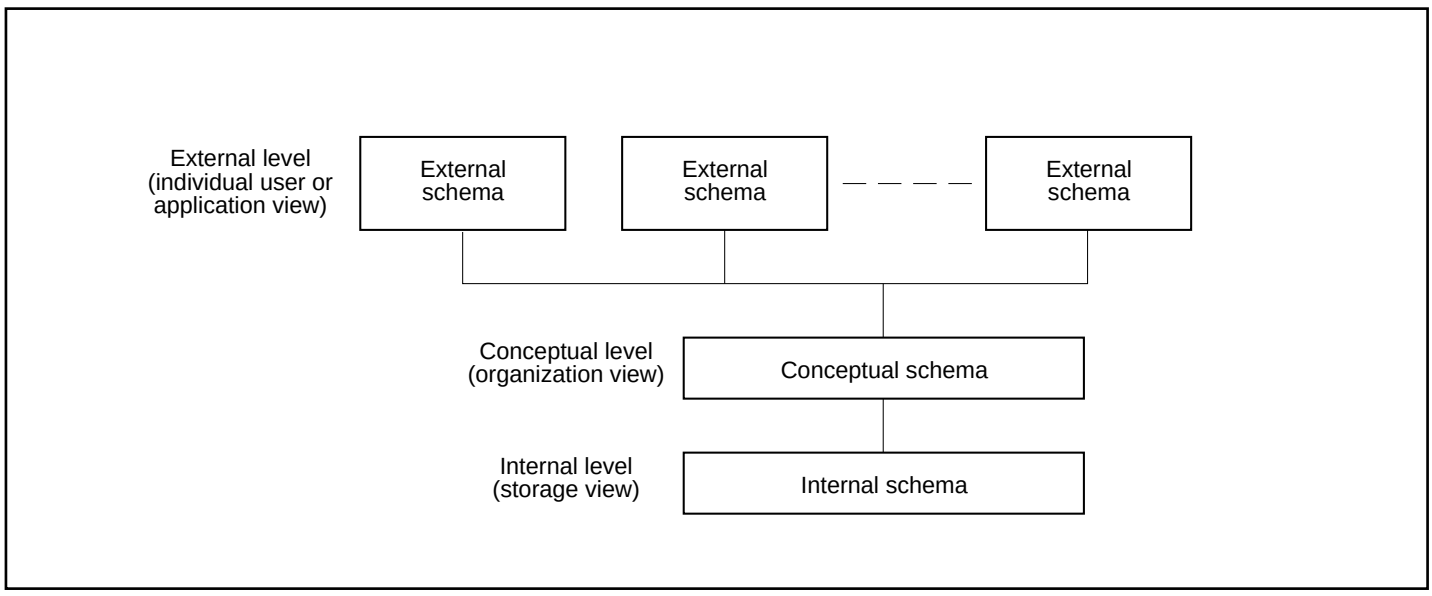
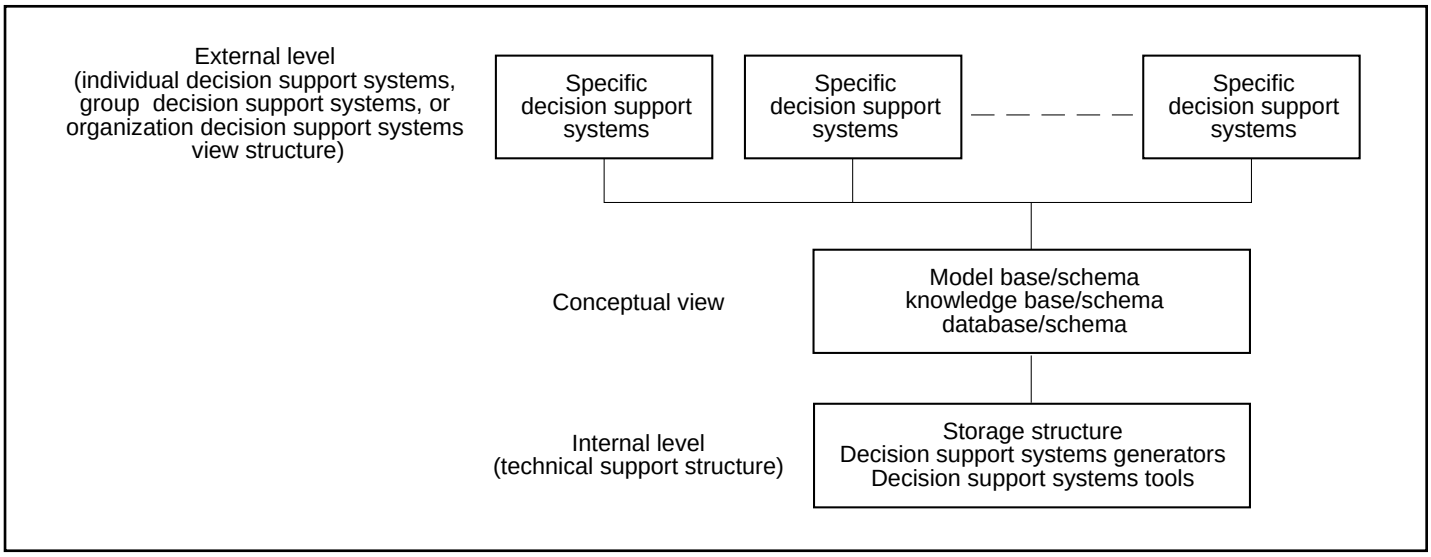


Figure 3. *A Three-level Integrated DSS Framework*

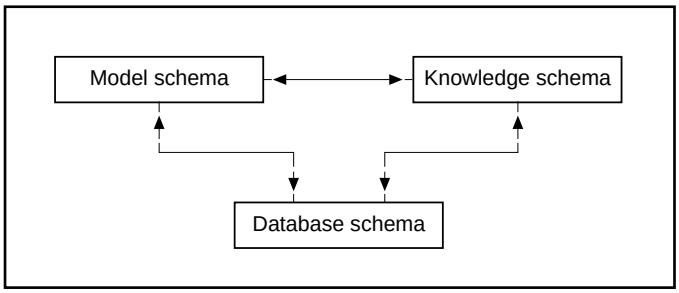


(as shown in Figure 5 ranging from MB-Module 1 to MB-Module *n*). Each MB-Module represents a generalized topic-based model. For instance, one MB-Module in the schema can represent financial plan models, while another MB-Module can represent marketing plan models, and so on (as shown in Figure 6). The reason for modularizing the structure of the model schema in the form of MB-Modules is to ensure proper organization and management of models.

An MB-Module will have links (relationships) with other MB-Modules in the schema. Using an artificial intelligence based (rule-based) form of model specification[25-28] each model can be represented as rules. The relationship between MB-Modules will exist if some clause in the body of a rule in one model (and MB-Module) is a rule in another model (and MB-Module). For example, consider sample rules in a marketing plan MB-Module (for a particular product) as follows:

IF Average selling cost per week = 600 AND
Sales effort = 100% AND
Selling price = 950
THEN Average quantity sold = 4

Figure 4. *The Conceptual View: Three Sets of Interacting Schemata*



IF Average quality sold > 4 AND
Sales effort = 100% AND
Selling price = 950
THEN Commission rate = 6%
ELSE Commission rate = 5%.

Further, suppose there is a sample rule in a financial plan MB-Module (for a particular product) as follows:

Figure 5. *Model Schema of Interrelated Modules*

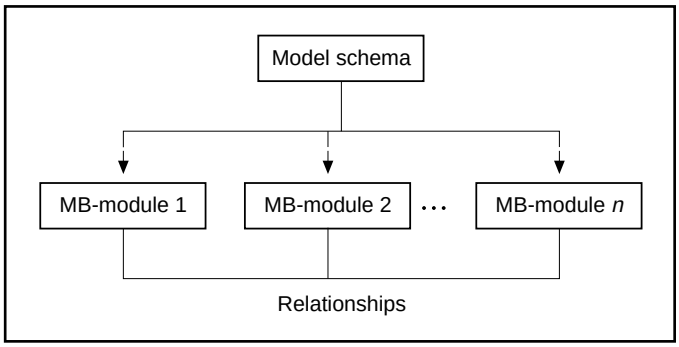
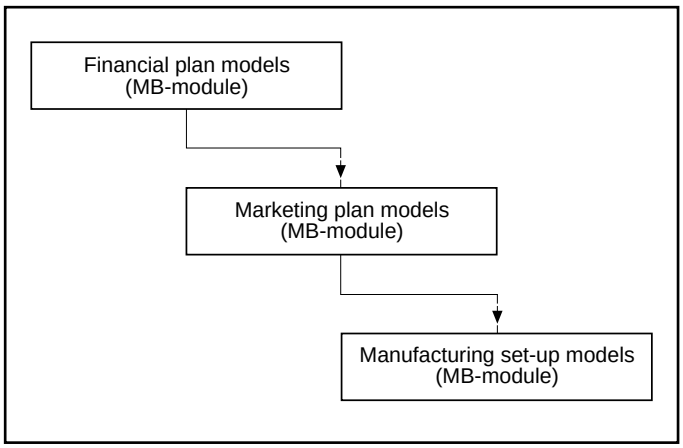


Figure 6. *Topic-based Model of Interrelated Modules*



IF Average quantity sold > 4 AND
Selling price = 950
THEN Revenue per week = (Average quantity sold × selling price).

Now, in the above example, the average quantity sold clause in the financial plan MB-Module rule is "related" to the rule in marketing plan MB-Module, where its basic value is determined. Preservation of such relationships will be essential for the integrity of various models in the model schema.

Knowledge Schema

The knowledge schema will be the logical repository of all knowledge in the form of generalized modules called KB-Modules (as shown in Figure 7, ranging from KB-Module 1 to KB-Module *n*). Each KB-Module is associated with one or more MB-Modules. Each KB-Module will contain knowledge:

(1) pertaining to assumptions, modelling tactic or general working direction;

(2) to facilitate input to models in the model base (schema); and

(3) to facilitate output from models in the model base (schema)[29-31].

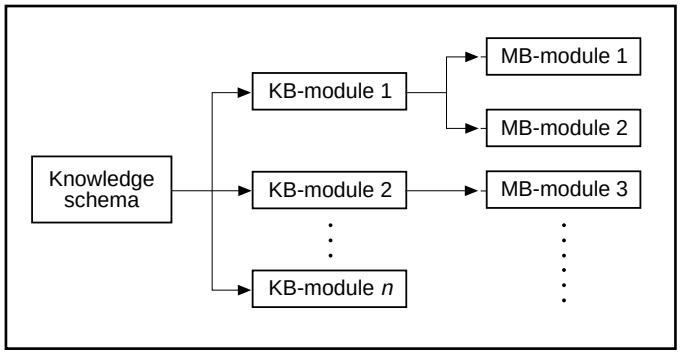
Sharing of a KB-Module will arise if different MB-Modules use similar knowledge. Using a rule-based knowledge representation schema, sample knowledge base rules in a KB-Module attached to the marketing plan MB-Module are as follows:

IF Commission rate = 6% AND
Sales effort = 100%
THEN Sales rep performance = Excellent
DISPLAY "Sales rep is meeting target sales"
IF Commission rate < 6% AND
Sales effort = 100%
THEN Sales rep performance = Good
DISPLAY "Sales rep should change strategy to meet target sales".

Database Schema

The database schema will be the repository of all data in the form of entities (or objects) needed to support DSS activity. The existing database schema of the organization may be utilized for supporting DSS development through the framework. If some entities necessary for DSS development are not included in the organization's database schema, then they need to be included in the framework's database schema.

Figure 7. *The Knowledge Schema of Modules*



External View

The external view is a representation of the model, data and knowledge needs of the specific DSS of different DSS. Any DSS represented by its specific DSS structure interacts with its related view structure. This is a change from the traditional DSS environment in that the different DSS (in the proposed integrated environment) will not have exclusive model base, database, and knowledge base components. The dialogue manager acts as a user interface to handle and control the semantics of interaction, as well as provide the linking of the user's semantics with the system's internal modelling, data and knowledge requirements as expressed in the view structure. Using an artificial intelligence/expert systems approach, the dialogue manager's interaction with a DSS specific view structure is shown in Figure 8.

The external view structure defines a user's specific DSS needs in the form of the models, knowledge and data components. Given the structure of the conceptual view, the external view structure specification consists of:

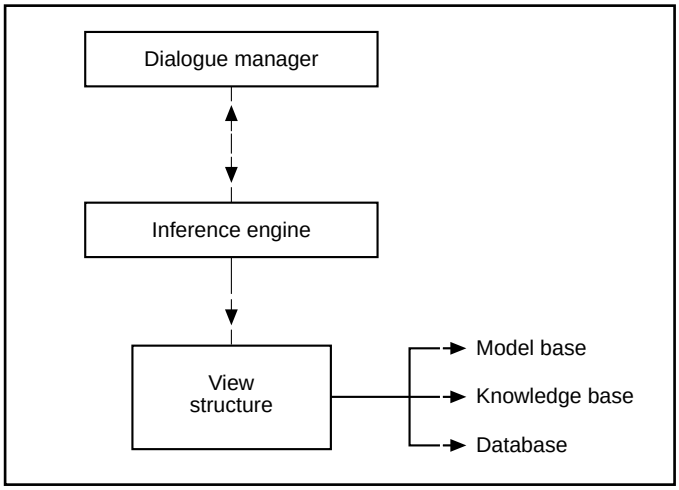
(1) models from MB-Module(s);

(2) knowledge from associated KB-Module(s); and

(3) database entities to support the model and knowledge components' working.

The models required for a specific-DSS will come from an individual MB-Module or from a number of MB-Modules. The attached inference engine will interact and control the working of the model, knowledge, and data components of the view structure. The inference engine

Figure 8. *The Interaction of the Dialogue Manager with a DSS Specific View Structure*



will further provide features like *why, how*, and other forms of explanation capability.

As DSS are interactive systems, the user may experiment and perform various analyses (for instance *what-if* analysis) on the view structure. However, the user will not be allowed to alter the structure of the models, knowledge and data in the view structure. The structure of the model, knowledge, and data will be altered only at the conceptual level.

The different DSS use of the framework will be as follows:

(1) In IDSS development, the individual would have the view structure specified, with the models from a single or a small set of MB-Modules.

(2) In GDSS development, the facilitator would create the appropriate view structure, with the models from a single or a small set of MB-Modules.

(3) In ODSS development, the view structure would be utilized by the organizational decision maker to integrate the criteria and models across a number of MB-Modules.

Internal View

The internal view would represent the technical specifications for the storage of the schemata in the conceptual view. Technical specifications include software and storage structures for maintaining the developed schemas. It could also include specifications for developing the schemas in the form of DSS-Generators or DSS-Tools.

Implementation Approach

The specification and maintenance of the three levels will be done by a DSS Management System (DSSMS). Similar to the ANSI/X3/SPARC database framework, there will be two levels of mappings in the integrated DSS framework. These mappings will be supported by the DSSMS. One level of mapping called external/conceptual mapping, will exist between the external view and the conceptual view. Another level of mapping called conceptual/internal mapping, will exist between the conceptual view and the internal view. Mappings would allow similar models, data, and knowledge components to be represented in different forms at the various levels, like different names or storage structures. For instance, it is possible for the external view structure entries to have different definitions from their related conceptual view entries.

Further, the two levels of mapping will make the conceptual view independent of a specific DSS structure in the external view, and will make the internal view changes independent of conceptual view changes. For instance, the external/conceptual mapping will make it possible to modify the conceptual view without altering the external view. The fitting of an external view to the conceptual view is through the related mapping. So when the conceptual view is modified, only the external/conceptual mapping that links an external view with the conceptual view needs to be modified. Similar effects will exist with respect to changes between the conceptual view and the internal view through the conceptual/internal mapping.

Some additional functions provided by the DSSMS would include:

(1) Query facility to query the conceptual view interactively.

(2) Screen and report generator to facilitate DSS application development.

(3) DSS schema dictionary of all details in the form of entries in the three schemata of conceptual view, external view structures, and internal view structures.

(4) DSS schema and external view definition facility.

The development of DSSMS is akin to the role of database management system (DBMS) software in organization. Whereas a DBMS supports database schema only, the DSSMS will support all the three schemata in the conceptual view. One approach towards implementation is to extend the knowledge base representation (as existing in expert system shells) to include both knowledge and model schemata[19], and link it with the database schema of a DBMS. However, any such approach will have to adapt the software environment further to provide the additional functions needed for DSSMS working.

Conclusions

The three-level DSS framework proposed in this article provides an initial structure to integrate DSS development in organizations. The framework resolves the redundancy of overlapping criteria and models; enables proper sharing of criteria and models; and provides specific DSS and support structure independence. An integrated environment ensures better relationships among decisions supported by different DSS.

Further research is in progress to extend the development and working of the framework. This involves:

(1) development of a general view structure in the form of a view screen;

(2) specifying appropriate storage structures pertaining to the internal view; and

(3) utilizing specific DSS of different DSS as a basis for studying the effects of the framework on DSS activity.

References

1. Sprague, R.H., "A Framework for the Development of Decision Support Systems", *MIS Quarterly*, Vol. 4, 1980.
2. Zachary, W., "A Cognitively-based Functional Taxonomy of Decision Support Techniques", *Human-Computer Interaction*, Vol. 2, 1986, pp. 25-63.
3. DeSanctis, G. and Gallupe, R.B., "A Foundation for the Study of Group Decision Support Systems", *Management Science*, Vol. 33, 1987, pp. 589-609.
4. Huber, G.P., "Issues in the Design of Group Decision Support Systems", *MIS Quarterly*, Vol. 8, 1984, pp. 195-204.
5 Kraemer, K.L. and King, J.L., "Computer-based Systems for Co-operative Work and Group Decision Making", *ACM Computing Surveys,* Vol. 20, 1988, pp. 115-46.
6. Pinsonneault, A. and Kraemer, K.L., "The Impact of Technological Support on Groups: An Assessment of the Empirical Research", *Decision Support Systems,* Vol. 5, 1989, pp. 197-216.
7. Bonczek, R.H., Holsapple, W.W. and Whinston, A.B., "Computer-based Support of Organizational Decision Making", *Decision Sciences*, Vol. 10, 1979, pp. 268-91.
8. Burns, A., Rathwell, M.A. and Thomas, R.C., "A Distributed Decision-making System", *Decision Support Systems*, Vol. 3, 1987, pp. 121-31.
9. George, J.F., "The Conceptualization and Development of Organizational Decision Support Systems", *Proceedings of the Twenty-Fourth Annual Hawaii International Conference on System Sciences*, Vol. III, 1991, pp. 57-64.
10. Hackathorn, R.D. and Keen, P.G.W, "Organizational Strategies for Personal Computing in Decision Support Systems", *MIS Quarterly*, Vol. 5, 1981, pp. 21-7.
11. Huber, G.P., "The Nature of Organizational Decision Making and the Design of Decision Support Systems", *MIS Quarterly*, Vol. 5, 1981, pp. 1-10.
12. Huber, G.P., "Effects of Decision and Communication Support Technologies on Organizational Decision Processes and Structures", in Lee, R.M., McCosh, A.M. and Migliarese, P. (Eds), *Organizational Decision Support Systems*, *Proceedings of the IFIP WG 8.3 Working Conference on Organizational Decision Support Systems*, Lake Como, Italy, June 1988, pp. 317-33.
13. Jarke, M., "Knowledge Sharing and Negotiation Support in Multiple-person Decision Support Systems", *Decision Support Systems*, Vol. 2, 1986, pp. 93-102.
14. Kaula, R. and Dumdum, U.R., "Towards an Organization DSS Architecture: An Open-Systems Perspective", *DSS-91 Transactions*, 1991, pp. 168-76.
15. King, J.L. and Star, S.L., "Conceptual Foundations for the Development of Organizational Decision Support Systems", *Proceedings of the HICSS-23 Conference*, January 1990, pp. 143-51.
16. Philippakis, A.S. and Green, G.I., "An Architecture for Organization-wide Decision Support Systems", *Proceedings of the Ninth ICIS*, Minneapolis, 1988, pp. 257-63.
17. Scher, J.M., "Distributed Decision Support Systems for Management and Organizations", *DSS-81 Transactions*, 1981, pp. 130-40.
18. Swanson, B., "Distributed Decision Support Systems: A Perspective", *Proceedings of the HICSS-23 Conference*, 1990, pp. 129-36.
19. Keen, P.G.W. and Hackathorn, R.D., "Decision Support Systems and Personal Computing", *Sloan School of Management*, Working Paper 1088-79, Cambridge, MA, 1979.
20. Martin, J., *Managing the Database Environment*, Prentice-Hall, Englewood Cliffs, NJ, 1983.
21. Sprague, R.H. and Carlson, E.D., *Building Effective Decision Support Systems*, Prentice-Hall, Englewood Cliffs, NJ, 1982.
22. Date, C.J., *An Introduction to Database Systems*, Addison-Wesley, Reading, MA, 1990.
23. Tsichritzis, D.C. and Klug, A., "The ANSI/X3/SPARC DBMS Framework Report of the Study Group on Database Management Systems", *Information Systems*, Vol. 3, 1978, pp. 173-91.
24. Tsichritzis, D.C. and Lochovsky, F.H., *Data Models*, Prentice-Hall, Englewood Cliffs, NJ, 1982.
25. Basu, A. and Dutta, A., "AI-based Model Management in DSS", *Computer*, September 1984.
26. Binbasioglu, M. and Jarke, M., "Domain-specific DSS Tools for Knowledge-based Model Building", *Decision Support Systems*, Vol. 2, 1986, pp. 213-23.
27. Fedorowicz, J. and Williams, G.B., "Representing Modeling Knowledge in an Intelligent Decision Support System", *Decision Support Systems*, Vol. 2, 1986, pp. 3-14.
28. Minch, R.P., "Logic Programming as a Paradigm for Financial Modeling", *MIS Quarterly*, Vol. 13, 1989, pp. 65-81.
29. Courtney, J.F., Paradice, D.B. and Mohammed, N.H.A., "A Knowledge-based DSS for Managerial Problem Diagnosis", *Decision Sciences*, Summer, 1987, pp. 373-99.
30. Turban, E. and Watkins, P., "Integrating Expert Systems and Decision Support Systems", *MIS Quarterly*, June 1986.
31. Turban, E., *Decision Support and Expert Systems: Management Support Systems*, Macmillan, New York, NY, 1990.

Rajeev Kaula is Assistant Professor in the Computer Information Systems Department at the Southwest Missouri State University, Missouri, USA.