1986

METHODOLOGIES FOR DSS ANALYSIS AND DESIGN: A CONTINGENCY APPROACH TO THEIR APPLICATION

Michael J. Ginzberg
Case Western Reserve University

Gad Ariav
New York University

METHODOLOGIES FOR DSS ANALYSIS AND DESIGN: A CONTINGENCY APPROACH TO THEIR APPLICATION

Michael J. Ginzberg
The Weatherhead School of Management
Case Western Reserve University

Gad Ariav
Graduate School of Business Administration
New York University

ABSTRACT

The analysis and design of systems for decision support typically has taken a single methodology approach, ignoring the fact that DSSs vary in their scope, complexity and purpose. This paper examines three primary approaches to DSS analysis and design in order to begin formalizing a consistent framework for the selection of a DSS development methodology. The contingency framework emphasizes the conditions under which the various methodologies are appropriate and likely to be effective, and provides guidelines for matching decision situations with development approaches.

INTRODUCTION

The topic of analysis and design of systems for decision support has occupied DSS researchers and practitioners for much of the last decade (Bennett, 1983). Typically, efforts in this domain have focussed on a single methodology for DSS development, operating under the implicit premise that there is, or ought to be, only one such methodology. Writers in this area have time and again focused on the methodology.

What is lacking is a broader view which recognizes that different decision situations call for different approaches to DSS analysis and design, i.e., that when one attempts to develop a DSS, he or she has to select an appropriate methodology from the set of available DSS development methodologies, and that the primary basis for this selection should be the salient characteristics of the decision situation under study. Such a pluralistic view is compatible with the more recent recognition that DSSs are varied in their scope, complexity and purpose.

This paper consolidates much of the prior work in this area by proposing an organizing framework for DSS analysis and design methodologies. The proposed organization of methodologies is explicitly usage-oriented. It emphasizes the conditions under which the various representative methodologies are appropriate and likely to be effective, and provides guidelines for matching decision situations and development approaches. As such, this paper adopts and advocates a more careful approach to the design of DSS development efforts.

Ariav and Ginzberg (1985) focused on the internal structure and architecture of DSSs. The current paper could be viewed as a natural sequel. This paper recapitulates briefly the notion of the DSS product and relates it to the DSS process, emphasizing the task of DSS design. It outlines three major approaches currently associated

with DSS design. Following their presentation, It presents a contingency approach for matching methodologies to situations, then discusses the implications of the proposed approach.

THE PRODUCT AND PROCESS OF DSS DESIGN

Product oriented definitions of DSS tend to emphasize the unique shape of computerized systems for decision support. Within this group of definitions, some emphasize the services rendered by the systems, while others focus on the set of components from which the systems are constructed. Ariav and Ginzberg (1985) suggest another, more balanced approach that combines these two. Based on the systemic premise, understanding decision support systems entails the simultaneous consideration of five aspects: environment, role, components, arrangement of components, and resources required to support the system (Churchman, 1968).

Tables 1 through 4 summarize four of the five aspects addressed within the systemic framework (a more complete discussion is the subject of Ariav and Ginzberg (1985)).

Table 1.

Task Characteristics
- Structurability
- Level
  - Operational Control, Management Control, or Strategic Planning
- Decision Process Phase
  - Intelligence, Design or Choice
- Functional Area
  - e.g., Finance, Marketing, Production

Access Pattern
- Mode of User Interaction
  - e.g., Online Dialog, Intermittent Batch
- User Community
  - e.g., Size, Expertise, and Role in the Decision Process
- Relationship to Neighboring IS

Table 2.

Levels of Support
- e.g., Retrieving and Displaying Raw Data, Suggesting or Selecting Solutions

Decision Range
- Generalized Versus Particularized

Supported Process
- e.g., Enhancing Individual Cognitive Processing Capabilities, Facilitating Learning, Supporting Communication and Coordination, Excercising Control

Table 3.

Dialog Management
- User Interface
- Dialog Control
- Request Generator

Data Management
- Query Language Facility
- Data Dictionary
- Database and DBMS
- Staging Facility

Model Management
- Modeling Command Processor
- Model-base and MBMS
- Model Execution
- Data Interface

Table 4.

Hardware

Software
- General Purpose Programming Languages
- DSS Tools
- DSS Generators
- Generalized DSS

People

Data

The fifth aspect of a system, arrangement of components, concerns the linkages among the DSS components, the nature of these linkages, and the justification for each of the linkages. In general, this aspect of DSS architecture has been treated in an extremely limited fashion in the literature.

A meaningful DSS design must explicitly link these five aspects so that characteristics of the system's environment and role will be reflected in its components and their arrangement, which in turn will become the basis for an intelligent selection of resources.

The process view of DSS recognizes a cycle or a loop which includes three elements: (1) the task(s) performed by a user(s), (2) the development activity, and (3) the delivered system (Figure 1). The task to be accomplished defines the capabilities needed in a DSS (Arrow 1). The design process translates these requirements into a working system (Arrow 2). Use of that system affects the task (Arrow 3) which alters the capabilities required and begins the cycle anew (Arrow 1). One can argue that all computer-based system development follows this basic loop pattern, and therefore the essential difference between MIS and DSS development is not in the nature of the cycle itself but in the velocity of movement within it. MIS development follows the same general cycle as DSS development, but at a much less rapid pace. As a result, the emphasis in the traditional design literature was on the development activity itself rather than on the whole loop.

The DSS literature has emphasized the value of this process, going so far as to argue that in some cases the process itself is the major source of benefits from the DSS concept (Hurst, *et al.*, 1983). The rationale behind this argument is that by applying systematic attention to a decision situation, the unstructured nature of that decision will necessarily decrease, making the process more understandable and controllable. In some sense, it was argued, any DSS is obsolete by the time it is built; therefore the tangible product is the added understanding that has been gained during the analysis and design process (Courbon, *et al.*, 1978).

The focus of this paper, the development component of the cycle, includes three subcomponents:

1. Analysis -- studying the salient attributes of the environment and defining the functional scope of and requirements for the proposed system;
2. Evaluation -- setting standards and developing procedures for monitoring and assessing system use, impact and performance; and
3. Construction -- translating requirements and specifications first into a general design and later into a specific plan of resource use (i.e., the implemented architecture).

The distinctions among these subcomponents are not necessarily sharp. Where does evaluation end and analysis start? Where does analysis end and construction start? Nevertheless, every approach to DSS design has to address all three elements and suggest ways to accomplish them. In general we require that the result of the systems analysis effort, i.e., the concise description of a DSS environment, be design-relevant, highlighting only those environmental features that have, or should have, an impact on the system structure. In the next section, we briefly present characteristic approaches to systems analysis for DSS. The discussion of these methodologies occupies the better part of a later section where a general framework for methodology selection is proposed.

OVERVIEW OF APPROACHES TO THE "DSS PROCESS"

A recent survey has shown that a wide variety of approaches to DSS development are used in practice (Hogue and Watson, 1984). It is especially interesting to note that among the 18 cases of successful systems included in the survey, 6 were developed in a manner similar to traditional MIS development and over periods of time ranging up to two years. This finding is contrary to conventional wisdom and suggests that the "quick and dirty" strategy is not the only successful approach to DSS development. Indeed, the repertoire of DSS design methodologies must be considered to include at least prototyping (Keen, and Gambino, 1983), the ROMC approach (Sprague and Carlson, 1982), and the decision research approach (Stabell, 1983).

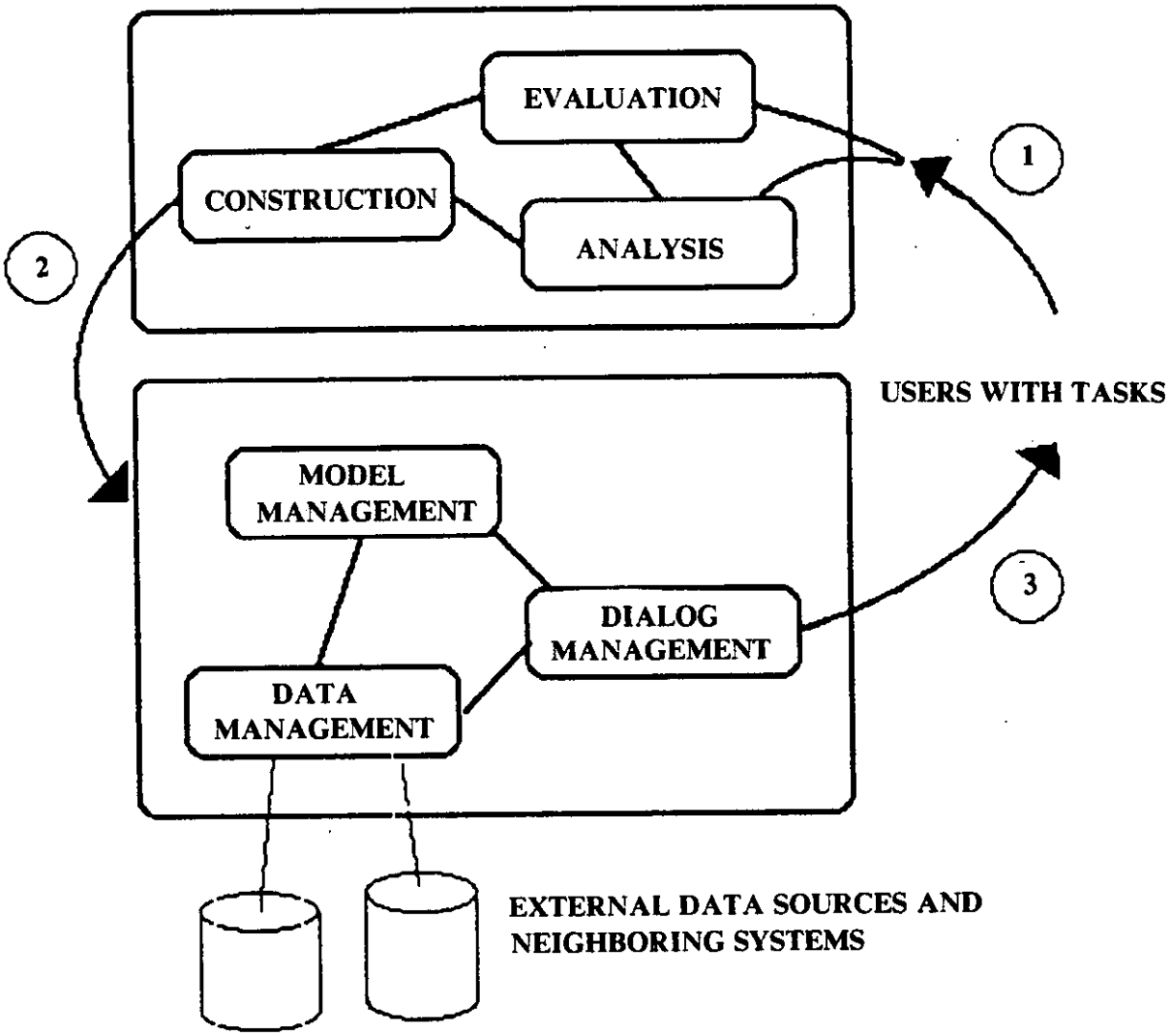


Figure 1: The DSS Process

In the remainder of this section we highlight the features of these three approaches to DSS systems analysis and identify the fundamental differences among them. The methods included in our review are those for which there is reasonable documentation and practical reference in the DSS literature. Methods that focus on narrow aspects (e.g., eliciting user preferences for information cues) typically stop short of discussing system design implications, and are not considered here.

Prototyping Approach

Prototyping has been traditionally associated with DSS design, and is still considered by many as the "ideal" development strategy for these systems. In our teaching and consulting experience we have found substantial resistance to even considering other approaches; to many, DSS and prototyping are synonymous. We use the term "prototyping" to describe a variety of essentially similar approaches which have been proposed for DSS design; e.g., middle-out design (Ness, 1975), the evolutive approach (Courbon, et al., 1978), and adaptive design (Keen, 1980).

One of the more rigorous prescriptions for the prototyping approach to DSS design is the "Linguistic Approach" (Keen, 1983). The process suggested by this approach starts with the identification of a "good" user; one who is knowledgeable about the decision problem and curious about the application of the new approach, will take initiative in the process, and enjoys the role of innovator.

In general, the focus is on the dialog, defining what the user says and sees. This becomes the conceptual model around which the system is developed. Data management and models should be dealt with only after this shell has been successful. Specifically, the designer should identify relevant verbs, both generic and special, that users naturally employ in their decision process. This may, for instance, tap the general financial jargon (e.g., "NPV"), and add some special mortgage terms (e.g., "balloon" or "cap"). Verbs should be translated into system commands, and care should be taken that all system commands match the users' verbs. The emphasis is on assuring that the system will correspond to the users' perception of the process.

Following this analysis, priorities are set for "Version 0," the first demonstration of the system to its potential users. This is partly a technical and partly a marketing decision. The designer has to identify the system capabilities that can be delivered quickly and cheaply, but will also be readily acceptable to the user. Version 0 must demonstrate feasibility and value. Adequate support of the users must accompany the physical Version 0 system.

From the base of Version 0, evolution to a more complex DSS should occur as user understanding of and demands made upon the system grow. There are differences between this linguistic approach and other prototyping approaches, but from our perspective they are minor. The essential features of all the prototyping methods as approaches to DSS design are: (1) focus initially on a small piece of the decision problem; (2) focus on the dialog, the user interface; (3) keep the initial system close to the user's existing decision process; (4) evolve and grow the system as the user's changing needs demand it.

ROMC Approach

The ROMC approach (shorthand for Representations, Operations, Memory aids, and Control (Sprague and Carlson, 1982)) broadens the range of concerns to be attended to during the analysis stage. As Sprague and Carlson describe it, the methodology oscillates between design and analysis, with no clear delineations between the two.

In essence, the methodology provides a check-list of issues and some coarse measures of internal consistency and closure. The four building blocks -- representations, operations, memory aids, and control -- define the elements of the check-list, and the relationships among them provide the consistency check. Representations capture the context in which users interpret outputs and invoke operations. As such, they should include the most natural images that are used during the decision process, and with which the decision maker is comfortable and familiar. For instance, in the environment of economic analysis, curves and equations are common and probably are absorbed by participants with minimal mental effort.

Operations are the ways users manipulate the images and objects of their decision environment. Continuing the economic analysis example, operations could include the invocation of a graph, scaling it up or down to highlight salient features, windowing and zooming on particular areas, performing a comparative overlay, and the like. One check on internal consistency (a clear requirement from any solid methodology!) is that operations operate on representations. Thus, an operation on an unspecified image identifies an incomplete representation. Likewise, a representation not manipulated by any operation indicates a problem with the specification.

Memory aids support the representations and operations by capturing the data that underlie their use. For instance, the content of a familiar form has to be analyzed and recorded, the data needed for a zoom operation have to be stored and examined, and a variety of related data sources are likely to be consulted throughout the decision making process (e.g., directories, personal notes and files, libraries).

Control provides the framework for integrating representations, operations, and memory aids into a useful decision making session. Designing the control entails examining and analyzing the context management in a typical decision session. The flow of the process may range from tightly controlled (there is a recommended way to go about the decision) to loosely controlled or user driven (the user has unrestricted access to the system's functions, and there are minimal restrictions on the order of execution). Procedures which are frequently repeated need to be captured and programmed. A related issue is that of assistance -- how much help does the typical user need? Design of a help structure is complementary to the session flow discussed above.

Since the building blocks of the method (Rs, Os, Ms, and Cs) are intimately interwoven, they serve as checkpoints for each other. If specification is complete, no loose ends should remain; all representations should be invocable, the context (control structure) for each operation should be specified, and every image should have its data available.

The ROMC analysis takes place as a precursor to design and construction. It is completely user oriented, and not at all oriented to hardware or software resources. It defines the scope of the decision process to be supported, but does not define the actual capabilities of the eventual system. The ensuing DSS construction should be based on it and should try to respond to it, but the approach itself provides no clear guidelines for translating functional requirements into a specific DSS design.

Decision Research Approach

The Decision Research approach (Stabell, 1983) focuses on the decision as the core of the process to be supported. The initial activity in this approach is to delineate the scope of the situation to be studied (goals, decision variables, environmental variables, etc.), which implies a choice of the specific decision situation to be supported.

In the next stage data are collected to support descriptive and normative modeling of the situation. Descriptive modeling attempts to describe the decision situation as it is. Data could be collected through structured observations, role episodes, "thinking aloud" protocols, or even regression approaches to associate problem stimuli with decision responses. Normative modeling describes the decision situation as it should be. Normative models may be based on literature review or expert opinion about the decision situation. A basic tenet of decision research is to use multiple models and data collection techniques to gain an understanding of the decision situation.

The descriptive and normative models of the decision process are the basis for the ensuing diagnosis of the problems in the decision process. Departures of the actual process from the normative process suggest areas for change, which might be brought about through the DSS. In that sense, discovering the weak spot of the decision process indicates what the DSS designer should focus on, and provides the functional specification for the DSS. For example, if a lack of attention to uncertainty has been diagnosed as a key shortcoming of the process, the DSS may incorporate probability density functions instead of a representative values (e.g., means and medians).

The Decision Research methodology is robust in the scientific sense, it is structurally "complete," and follows a very clear and appealing logic. It also involves a major analysis effort up-front, and in its scope resembles the traditional (i.e., SDLC or "MIS") effort of system analysis.

Summary of Methodologies

We can summarize and compare these three approaches by mapping them onto the analysis, evaluation and construction activities which comprise the development process (Table 5).

In the prototyping approach, analysis is limited to the identification of verbs, a "good" user is substituted for an explicit evaluation procedure, and the major burden is on the actual construction as the basis for evaluation. In the ROMC approach there is a clearer analysis element, with fairly clear guidelines for establishing correspondence between the findings of the analysis and the three generic functions of the DSS. There is, however, no explicit basis for evaluation. In the decision research approach, there is a major emphasis on the analysis activity, and the results of the analysis point clearly to appropriate criteria for evaluation. This approach provides no obvious way to translate analysis findings into detailed designs.

SELECTING A DESIGN METHODOLOGY: A CONTINGENCY FRAMEWORK

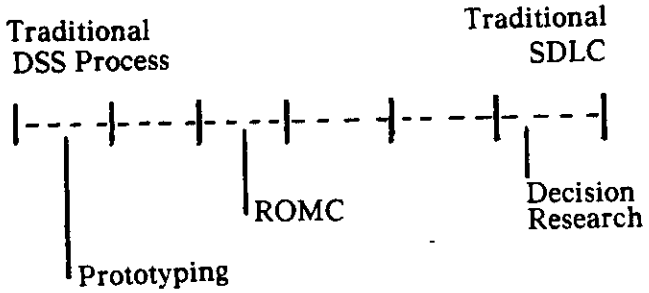
The basic premise of the contingency framework outlined in this paper is that each methodology emphasizes or attends to specific aspects of DSS development situations. Further, each methodology can be characterized with respect to the key dimensions of the DSS environment and role outlined in a previous section. Hence, the core of the contingency approach to methodology selection is the fact that methodologies can be mapped onto a continuum. They should not be viewed as competing with each other, but rather as complementary; each methodology responds best to a different set of environmental conditions. This basic idea is depicted in Figure 2.

Moving from left to right on this continuum generally implies (1) more effort (e.g., time) spent "up-front"; (2) decreasing "respect" for the current decision making process; (3) an increasingly normative approach; (4) increasing attention to ultimate "decisional impact," i.e., change in actual performance; and, as a result, (5)

Table 5. Design Sub-Tasks in DSS Design Approaches

Sub Tasks	Proto-Typing	ROMC	Decision Research
Analysis	verbs and dialog only	problem representation; clear object classes; consistency checks	major focus; well defined process for analysis
Evaluation	"good" user substitutes for explicit criteria	no explicit basis	explicit part of process; based on diagnosis for decision process change
Construction	major part of effort; built around verbs	reasonably direct translation from R. O. M. & C definitions	no explicit consideration or mechanisms

Figure 2. Mapping of DSS Design Methodologies.



potentially more radical impact on the decision situation and outcomes.

In the remainder of this section we relate these various methodologies to the design relevant aspects of the DSS Environment and Role developed in (Ariav and Ginzberg, 1985). At the moment these scattered observations are based on anecdotal evidence and experience. More rigorous research is needed to validate and extend the framework.

Task structurability. Since prototyping in itself adds no structure to the situation (that is, it mirrors the existing process), it seems to be most appropriate in two extreme situations: (1) where substantial structure has already been applied to the task and the user's view of the appropriate decision process is likely to be stable, or (2) in the complete opposite situation, where structure is (at least for the moment) so elusive that any attempt to tackle it directly seems to be futile. In general, the more structurable the task the more applicable decision research seems to be.

Task level. Task level by itself should not favor one or another of the design methodologies. However, to the extent that operational control tasks are more structurable and strategic planning tasks less so, decision research is likely to be more appropriate for the former and prototyping for the latter.

Decision process phase. Support of the design phase requires a "theory" of how alternatives are created in the particular decision domain. Neither prototyping nor ROMC provide such a theory, but the normative modeling aspect of decision research does.

Functional area. There is no reason to expect any design methodology to be more appropriate to one functional area than to another.

Modes of user interaction. The paramount importance of dialog compatability in highly interactive environments suggests that these situations may be more effectively handled with prototyping (or ROMC) than with decision research. One must be careful, of course, not to fall into what Stabell (1983) terms "the usability trap," that is, a system that is usable and used, but not useful.

User community. Several aspects of the user community should bear on methodology selection. The methodologies differ in the degree to which they depend on the user. The more structured the method (the further to the right on the continuum) the less it depends on the user, and the more is assumed to be done by the analyst. Thus, in situations where users have great problem area expertise, prototyping may be quite appropriate, while decision research would be more appropriate if users have relatively little expertise. Another important dimension is the size of the user base. A smaller user group is more amenable to the less structured methodology, while a large user group virtually requires a more structured approach. This might explain why prototyping is so much associated with DSS today, which is still largely individually oriented. User proficiency and frequency of use are also factors in selecting the methodology; i.e., the less structured ones depend on a committed and knowledgeable user, and therefore may not be appropriate for systems to support a casual user.

Relationship to other computer-based systems. DSS which must interact with "neighboring systems" would seem less amenable to prototyping. The interactions among the various systems have to be specified at a very detailed level, and typically are not handled effectively on a trial and error basis. The growing trend toward linked systems provides an additional reason for considering more structured approaches, which entail more careful planning.

Level of support. Clearly the higher one moves in Alter's taxonomy (Alter, 1980), i.e., the more model intensive the system is intended to be, the more up-front effort is required. This suggests that a more structured methodology (i.e., decision research) will be appropriate for these systems. Less model-intensive support, i.e., raw data retrieval, statistical analysis, and simple modeling can be handled by less structured methods, e.g., prototyping or ROMC.

Decision range. DSS intended to support a wide range of decision makers and situations are those most in need of a normative approach to decision making. Such systems cannot be tailored to an individual's existing decision process, since there is no single decision maker whose process can be modeled. Thus, decision research would appear to be the most appropriate methodology for such systems. There is, however, a problem. The more generalized the system, the less likely it is that meaningful normative decision models exist, and some other approach (e.g., ROMC) may be most readily applied. The GADS system would seem to be an instance of this type.

Supported process. All decision related processes -- cognitive support, learning facilitation, and communication/coordination support -- would seem amenable to all three development methodologies.

DISCUSSION AND IMPLICATIONS

Perhaps the clearest implication of the contingency approach to design methodology selection is that it makes the issue of methodology choice explicit, and makes the designer aware of the ramifications of the choice; e.g., choosing prototyping implies a heavy reliance on users, fast pace of system construction, but slow impact on the decision situation. Making a choice of methodology "forces" the designer to consider, at the outset, what problems he is trying to address in building the DSS. The problematic content of the DSS (e.g., being potentially highly unstructured), should not rule out a judicious selection of an "appropriate" design methodology. Arguing about the strategy for attacking complexity is meaningful even though the complexity itself may be very unyielding.

The selection of a development methodology, as suggested in this paper, follows the systemic "outside-in" approach -- it follows from an understanding of the environment of the entire DSS development effort and its role. Unfortunately, the literature on DSS design has provided only limited guidance for an environmentally based selection of design methodology. A change from touting the DSS methodology to explicit selection and justification of a methodology is needed in DSS research. Such a change does not imply that studies of a single methodology should be avoided, only that in any study we must consider the methodology as coming from a spectrum of possible methods, and assess its relative merits and disadvantages.

With the introduction of the contingency framework for design methodologies, attention shifts from the mere introduction of design approaches and methodologies to a broader analysis of "conditions for adequacy." Research is needed to establish the causal basis for the contingencies, and to develop or refine methodologies for the empty area toward the right hand end of the continuum in Figure 2. Specifically, the potential of AI methods should be examined, as they address the fundamental thought processes involved in design. This is another piece in the DSS-AI puzzle that has to be sorted out.

Recognition of the fact that any methodology is but one of several possibilities also suggests a refocusing in DSS curricula. Too many DSS courses are built around a specific methodology. This kind of focus does not give students an understanding of the ramifications of their (typically) implicit choice, and they are therefore unaware of the limitations they have unknowingly assumed. An alternative (which we have used in our DSS classes and seminars) is to present the students with a range of methodologies, and ask them (as part of a term project) to justify their strategy of development. This helps them to focus on their objectives in introducing the DSS, or at least makes them aware of the various assumptions they have been holding about the decision situation to be supported.

Peter Keen and others have recently argued for a new focus in DSS research, a focus on "Decisions That Matter" (Elam, et al., 1984). Their point is that since we have "mastered" the technology of DSS design, we ought to focus our attention now on where to apply these systems for the greatest benefit. The shift is from the traditional concern with effectiveness of the specific system and efficiency in its realization to the broader, organizational view of effectiveness in the application of decision support.

The contingency framework presented here allows us to analyze the ramifications of this shift in focus on the DSS design process. As far as

the inventory itself is concerned, a front-end component will probably be added to current methodologies to support an identification stage. CSF (or some similar technique) (Henderson, 1985) is a likely candidate to be used as this front-end methodology.

In terms of relative emphasis within the methodologies, those with clearer evaluation aspects will be favored, as DSS will address more vital concerns. This argument leaves us with prototyping and decision research as major contenders. We might therefore expect more emphasis on decision research (and possibly other more normatively based approaches) in an attempt to more carefully study the problems to which DSS are applied.

On the other hand, under the assumption that decisions that matter are especially "wicked," the prototyping approach could remain dominant, since it allows one to begin work on a problem with less initial structuring than do the other methodologies. Another factor which supports the continued use of prototyping is the need for an individualized approach to DSS design when it deals with concerns that occupy higher levels in the organization.

CONCLUSION

Now that there is general agreement on the role of DSS and an accepted generic view of DSS architecture is emerging, understanding the nature of the process through which this general view is customized to fit particular combinations of environmental aspects, i.e., the analysis and design of DSS, is becoming increasingly important.

The framework presented in this paper is clearly only the beginning. Following the completion and preliminary evaluation of this proposed framework, a deeper experimental evaluation will be possible. Further evaluation and development of this "road map" of methodologies should examine its descriptive validity, i.e., the usefulness of the mapping in providing new insights about the state of research and practice of DSS development. A rudimentary attempt to do so was presented in the preceeding section.

Another immediate research concern is with the prescriptive validity of the framework, i.e., the extent to which the proposed scheme explains cases of DSS development problems by a mismatch between the characteristics of the decision situation and the methodology applied. Such problems have been identified in some DSS projects, and deeper analysis of these cases has to be performed in light of the differentiated conditions for the application of each methodology. Laboratory experiments and case studies are conceivable research approaches to the investigation of these issues.

Finally, analysis and design cannot really be considered apart from the broader question of implementation strategy. The integration of design and implementation in an environmental context is a needed next step.

Acknowledgement

We would like to thank Phillip Ein-Dor for his review of, and his comments on, a preliminary version of this paper.

REFERENCES

Alter, S. *Decision Support Systems: Current Practices and Continuing Challenges*, Addison-Wesley, Reading, Massachusetts, 1980.

Ariav, G. and Ginzberg, M.J. "DSS Design--A Systemic View of Decision Support," *Communications of the ACM*, Volume 28, Number 10, October 1985, pp.1045-1052.

Bennett, J.L. (ed.) *Building Decision Support Systems*, Addison-Wesley, Reading, Massachusetts, 1983.

Churchman, C.W. *The System Approach*, Dell Publishing Co., Inc., New York, New York, 1968.

Courbon, J.C., Grajew, J., and Tolovi, J. "Design and Implementation of Decision Support Systems by an Evolutive Approach," Technical Report, Grenoble, France, 1978.

Elam, J.J., Henderson, J.C., Keen, P.G.W., Konsynski, B., Meador, C.L., and Ness D. "A Vision for Decision Support Systems," unpublished manuscript, November 1984.

Henderson, J. "A Planning Methodology for Identifying Strategic Opportunities for DSS," in *Proceedings of the 1985 NYU Symposium on Integrating Systems for End-Users*, M. Jarke, (ed), New York, May 1985, pp.169-188,

Hogue, J.T., and Watson, H.J. "Current Practices in the Development of Decision Support Systems," in (eds.), *Proceedings of the Fifth International Conference on Information Systems*, J. King and R. Kramer (eds.), Tucson, AZ, November 1984, pp.117-128.

Hurst, E.G., Ness, D.N., Gambino, T.J., and Johnson, T.H. "Growing DSS: A Flexible, Evolutionary Approach," in *Building Decision Support Systems*, J. L. Bennett, (ed.), Addison-Wesley, Reading, Massachusetts, 1983, pp.111-132.

Keen, P.G.W. "Adaptive Design for Decision Support Systems," *Data Base*, Volume 12, Number 1 & 2, Fall 1980, pp.31-40.

Keen, P.G.W, and Gambino, T.J. "Building a Decision Support System: The Mythical Man-Month Revisited," in *Building Decision Support Systems*, J. L. Bennett (ed.), Addison-Wesley, Reading, Massachusetts 1983, pp.133-172.

Ness, D.N. "Theories of DSS Design," Technical Report, Department of Decision Sciences, University of Pennsylvania, February, 1975.

Sprague, R.H., and Carlson, E.D. *Building Effective Decision Support Systems*, Prentice-Hall, Englewood Cliffs, New Jersey, 1982.

Stabell, C.B. "A Decision-Oriented Approach to Building DSS," in *Building Decision Support Systems*, J. L. Bennett (ed.), Addison-Wesley, Reading, Massachusetts, 1983, pp.221-260.