A Framework for the Development of Decision Support Systems

By: Ralph H. Sprague, Jr.

Abstract

This article proposes a framework to explore the nature, scope, and content of the evolving topic of Decision Support Systems (DSS). The first part of the framework considers (a) three levels of technology which have been designated DSS, (b) the developmental approach that is evolving for the creation of a DSS, and (c) the roles of several key types of people in the building and use of a DSS. The second part develops a descriptive model to assess the performance objectives and the capabilities of a DSS as viewed by three of the major participants in their continued development and use. The final section outlines several issues in the future growth and development of a DSS as a potentially valuable type of information system in organizations.

Keywords: Decision Support Systems, development approach, performance objectives, capabilities, issues

ACM Categories: 3.3, 3.5, 3.6

Introduction

We seem to be on the verge of another "era" in the relentless advancement of computer based information systems in organizations. Designated by the term Decision Support Systems (DSS), these systems are receiving reactions ranging from "a major breakthrough" to "just another 'buzz word'."

One view is that the natural evolutionary advancement of information technology and its use in the organizational context has led from EDP to MIS to the current DSS thrust. In this view, the DSS picks up where MIS leaves off. A contrary view portrays DSS as an important subset of what MIS has been and will continue to be. Still another view recognizes a type of system that has been developing for several years and "now we have a name for it." Meanwhile, the skeptics suspect that DSS is just another "buzz word" to justify the next round of visits from the vendors.

The purpose of this article is to briefly examine these alternative views of DSS, and present a framework that proves valuable in reconciling them. The framework articulates and integrates major concerns of several "stakeholders" in the development of DSS: executives and professionals who use them, the MIS managers who manage the process of developing and installing them, the information specialists who build and develop them, the system designers who create and assemble the technology on which they are based, and the researchers who study the DSS subject and process.

Definition, Examples, Characteristics

The concepts involved in DSS were first articulated in the early '70's by Michael S. Scott Morton under the term "management decision systems" [32]. A few firms and scholars began to develop and research DSS, which became characterized as interactive computer based systems, which help decision makers utilize data and *mode/s* to solve *unstructured* problems. The unique contribution of DSS resulted from these key words. That definition proved restrictive enough that few actual systems completely

satisfied it. Some authors recently extended the definition of DSS to include any system that makes some contribution to decision making; in this way the term can be applied to all but transaction processing. A serious definitional problem is that the words have a certain "intuitive validity;" any system that supports a decision, in any way, is a "Decision Support System."

Unfortunately, neither the restrictive nor the broad definition helps much, because they do not provide guidance for understanding the value, the technical requirements, or the approach for developing a DSS. A complicating factor is that people from different backgrounds and contexts view a DSS quite differently. A manager and computer scientist seldom see things in the same way.

Another way to get a feeling for a complex subject like a DSS is to consider examples. Several specific examples were discussed in The Society for Management Information Systems (SMIS) Workshop on DSS in 1979 [35]. Alter examined fit-six systems which might have some claim to the DSS label, and used this sample to develop a set of abstractions describing their characteristics [1,2]. More recently, Keen has designated about thirty examples of what he feels are DSS and compares their characteristics [26].

The "characteristics" approach seems to hold more promise than either definitions or collections of examples in understanding a DSS and its potential. More specifically, a DSS may be defined by its capabilities in several critical areas-capabilities which are required to accomplish the objectives which are pursued by the development and use of a DSS. Observed characteristics of a DSS which have evolved from the work of Alter, Keen, and others include:

- they tend to be aimed at the less well structured, underspecified problems that upper level managers typically face;

*they attempt to combine the use of models or analytic techniques with traditional data access and retrieval functions;

*they specifically focus on features which make them easy to use by noncomputer people in an interactive mode; and

- they emphasize flexibility and adaptability to accommodate changes in the environment and the decision making approach of the user.

A serious question remains. Are the definitions, examples, and characteristics of a DSS sufficiently different to justify the use of a new term and the inference of a new era in information systems for organizations, or are the skeptics right? Is it just another "buzz word" to replace the fading appeal of MIS?

DSS Versus MIS

Much of the difficulty and controversy with terms like "DSS" and "MIS" can be traced to the difference between an academic or theoretical definition and "connotational" definition. The former is carefully articulated by people who write textbooks and articles in journals. The latter evolves from what actually is developed and used in practice, and is heavily influenced by the personal experiences that the user of the term has had with the subject. It is this connotational definition of EDP/MIS/DSS that is used in justifying the assertion that a DSS is an evolutionary advancement beyond MIS.

This view can be expressed using Figure 1, a simple organizational chart, as a model of an organization. EDP was first applied to the lower operational levels of the organization to automate the paperwork. Its basic characteristics include:

*a focus on data, storage, processing, and flows at the operational level;

- efficient transaction processing;

*scheduled and optimized computer runs;

*integrated files for related jobs; and

*summary reports for management.

In recent years, the EDP level of activity in many firms has become a well-oiled and efficient production facility for transactions processing.

The MIS approach elevated the focus of information systems activities, with additional emphasis on integration and planning of the information

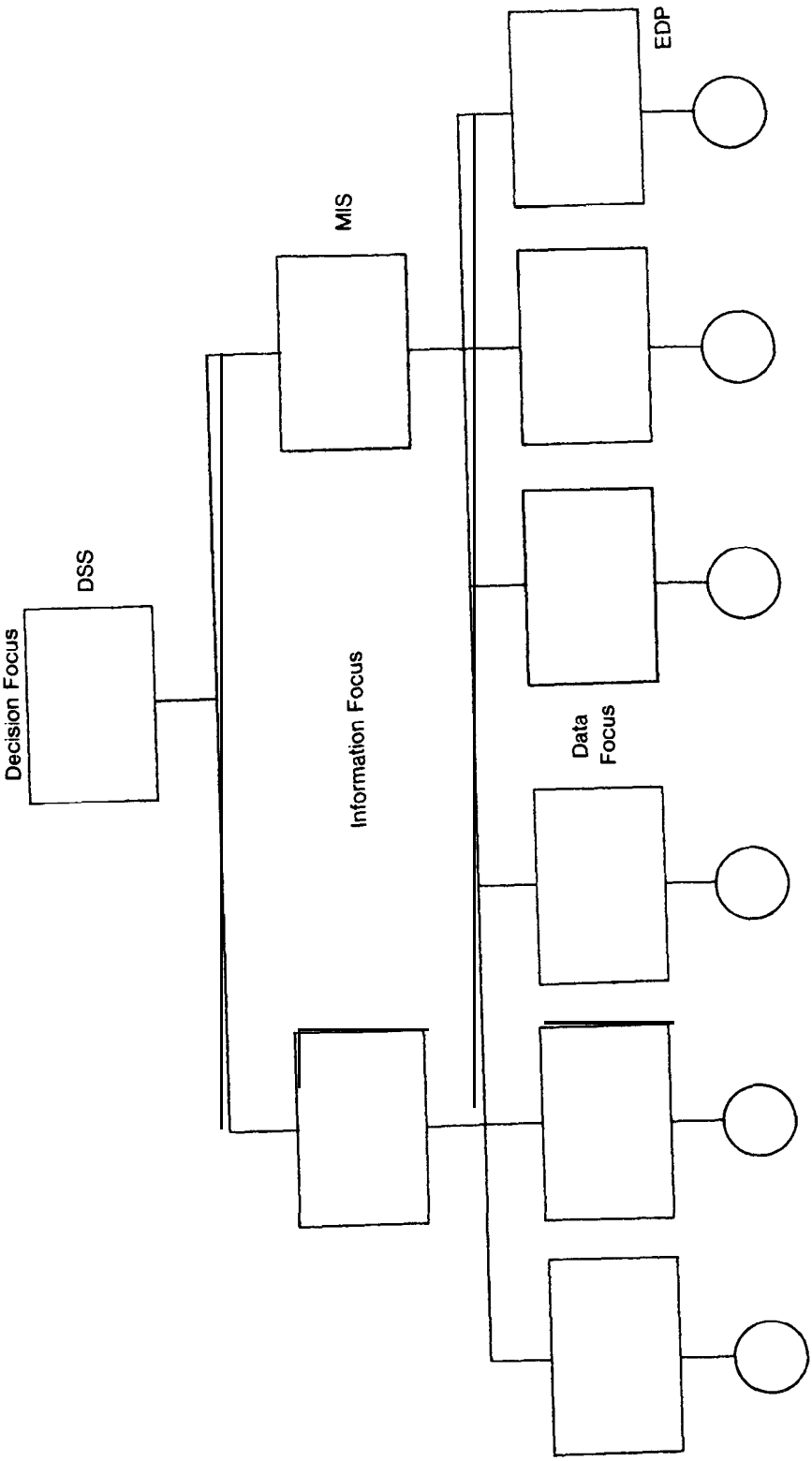

Figure 1. The Connotational View

systems function. In *practice,* the characteristics of MIS include:

*an information focus, aimed at the middle managers;

*structured information flow;

• an integration of EDP jobs by business function, such as production MIS, marketing MIS, personnel MIS, etc.; and

*inquiry and report generation, usually with a database.

The MIS era contributed a new level of information to serve management needs, but was still very much oriented to, and built upon, information flows and data files.

According to this connotational view, a DSS is focused still higher in the organization with an emphasis on the following characteristics:

*decision focused, aimed at top managers and executive decision makers;

*emphasis on flexibility, adaptability, and quick response;

• user initiated and controlled; and

• support for the personal decision making styles of individual managers.

This connotational and evolutionary view has some credence because it roughly corresponds to developments in practice over time. **A** recent study found MIS managers able to distinguish the level of advancement of their application systems using criteria similar to those above [27]. Many installations with MIS type applications planned to develop applications with DSS type characteristics. However, the "connotational" view has some serious deficiencies, and is definitely misleading in the further development of a DSS.

*It implies that **decision** *support* is needed only at the top levels. In fact, *decision support* is required at all levels of management in the organization.

*The decision making which occurs at several levels frequently must be coordinated. Therefore, an important dimension of *decision support* is the communication and coordination between decision makers across organizational levels, as well as at the same level.

• It Implies that **decision** support is the only thing top managers need from the information system. In fact, decision making is only one of the activities of managers that benefits from information systems support.

There is also the problem that many information systems professionals, especially those in SMIS, are not willing to accept the narrow connotationat view of the term "MIS." To us, MIS refers to the entire set of systems and activities required to manage, process, and use information as a resource in the organization.

The Threoretical View

To consider the appropriate role of a DSS in this overall context of information systems, the broad charter and objectives of the information systems function in the organization is characterized:

Dedicated to improving the performance of knowledge workers in organizations through the application of information technology.

*Improving the performance is the ultimate objective of information systems-not the storage of data, the production of reports, or even "getting the right information to the right person at the right time." The ultimate objective must be viewed in terms of the ability of information systems to support the improved performance of people in organizations.

*Knowledge workers are the clientele. This group includes managers, professionals, staff analysts, and clerical workers whose primary job responsibility is the handling of information in some form.

*Organizations are the context. The focus is on information handling in goal seeking organizations of all kinds.

- The application of information technology is the challenge and opportunity facing the information systems professional for the purposes and in the contexts given above.

A triangle was used by Robert Head in the late '60's as a visual model to characterize MIS in this broad comprehensive sense [22]. It has become a classic way to view the dimensions of an information system. The vertical dimension represented the levels of management, and the horizontal dimension represented the main functional areas of the business organization. Later authors added transactional processing as a base on which the entire system rested. The result was a two dimensional model of an MIS in the broad sense — the total activities which comprise the information system in an organization. Figure 2 is a further extension of the basic triangle to help describe the concept of the potential role of a DSS. The depth dimension shows the major technology "subsystems" which provide support for the activities of knowledge workers.

Three major thrusts are shown here, but there could be more. The structured reporting system includes the reports required for the management and control of the organization, and for satisfying the information needs of external parties. It has been evolving from efforts in EDP and MIS, in the narrow sense, for several years. Systems to support the communication needs of the organization are evolving rapidly from advances in telecommunications with a strong impetus from office automation and word processing. DSS seems to be evolving from the coalescence of information technology and operations research/management science approaches in the form of interactive modeling.

To summarize this introductory section, a DSS is not merely an evolutionary advancement of EDP

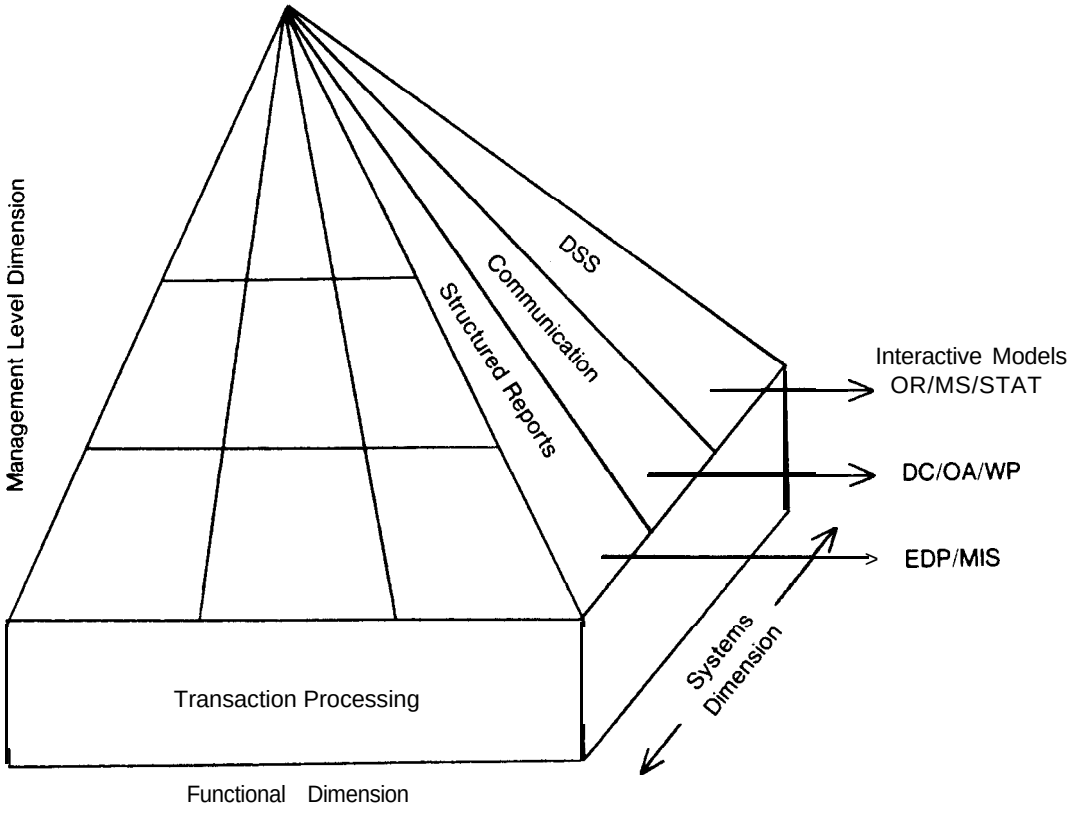

Figure 2. The **Complete View**

and MIS, and it will certainly not replace either. Nor is it merely a type of information system aimed exclusively at top management, where other information systems seem to have failed. A DSS is a class of information system that draws on transaction processing systems and interacts with the other parts of the overall information system to support the decision making activities of managers and other knowledge workers in the organizations. However, there are some subtle but significant differences between a DSS and traditional EDP or so-called MIS approaches. Moreover, these systems require a new combination of information systems technology to satisfy a set of heretofore unmet needs. It is not yet clear exactly how these technologies fit together, or which important problems need to be solved. Indeed, that is a large part of the purpose of this article. It is apparent, however, that a DSS has the potential to become another powerful weapon in the arsenal of the information systems professional to help improve the effectiveness of the people in organizations.

The Framework

The remainder of this article is devoted to an exploration of the nature of this "thrust" in information systems called "DSS." The mechanism for this exploration is another of the often maligned but repeatedly used "frameworks."

A framework, in the absence of theory, is helpful in organizing a complex subject, identifying the relationships between the parts, and revealing the areas in which further developments will be required. The framework presented here has evolved over the past two years in discussions with many different groups of people.' It is organized in two major parts. The first part considers: (a) three levels of technology, all of which have been designated as a DSS, with considerable confusion; (b) the developmental approach that is evolving for the creation of a DSS; and (c) the roles of several key types of people in the building and use of a DSS. The second part of the framework develops a descriptive model to assess the performance objectives and the capabilities of a DSS as viewed by three of the major stakeholders in their continued development and use.

Three technology levels

It is helpful to identify three levels of hardware/software which have been included in the label "DSS." They are used by people with different levels of technical capability, and vary in the nature and scope of task to which they can be applied.

Specific DSS

The system which actually accomplishes the work might be called the *Specific DSS.* It is an information systems "application," but with characteristics that make it significantly different from a typical data processing application. It is the hardware/software that allows a specific decision maker or group of decision makers to deal with a specific set of related problems. An early example is the portfolio management system [20] also described in the first major DSS book by Keen and Scott Morton [23]. Another example is the police beat allocation system used on an experimental basis by the City of San Jose, California [9]. The latter system allowed a police officer to display a map outline and call up data by geographical zone, showing police calls for service, activity levels, service time, etc. The interactive graphic capability of the system enabled the officer to manipulate the maps, zones, and data to try a variety of police beat alternatives quickly and easily. In effect, the system provided tools to *amplify* a manager's judgment. Incidentally, a later experiment attempted to apply a traditional linear programming model to the problem. The solution was less satisfactory than the one designed by the police officer.

DSS **Generator**

The second technology level might be called a *DSS Generator.* This is a "package" of related hardware and software which provides a set of capabilities to quickly and easily build a Specific DSS. For example, the police beat system described above was built from the Geodata Analysis and Display System (GADS), an experimental system developed at the IBM Research Laboratory in San Jose [8]. By loading different maps, data, menu choices, and procedures or command strings, GADS was later used to build a Specific DSS to support the routing of IBM copier repairmen [42]. The

development of this new "application" required less than one month.

Another example of a *DSS Generator* is the Executive Information System (EIS) marketed by Boeing Computer Services [6]. EIS is an integrated set of capabilities which includes report preparation, inquiry capability, a modeling language, graphic display commands, and a set of financial and statistical analysis subroutines. These capabilities have all been available individually for some time. The unique contribution of EIS is that these capabilities are available through a common command language which acts on a common set of data. The result is that EIS can be used as a DSS Generator, especially for a Specific DSS to help in financial decision making situations.

Evolutionary growth toward DSS Generators has come from special purpose languages. In fact, most of the software systems that might be used as Generators are evolving from enhanced planning languages or modeling languages, perhaps with report preparation and graphic display capabilities added. The Interactive Financial Planning System (IFPS) marketed by Execucom Systems of Austin, Texas [18], and EXPRESS available from TYMSHARE [44], are good examples.

DSS Tools

The third and most fundamental level of technology applied to the development of a DSS might be called *DSS* Tools. These are hardware or software elements which facilitate the development of a specific DSS or a DSS Generator. This category of technology has seen the greatest amount of recent development, including new special purpose languages, improvements in operating systems to support conversational approaches, color graphics hardware and supporting software, *etc.* For example, the GADS system described above was written in FORTRAN using an experimental graphics subroutine package as the primary dialogue handling software, a laboratory enhanced raster-scan color monitor, and a powerful interactive data extraction/database management system.

Relationships

The relationships between these three levels of technology and types of DSS are illustrated by Figure 3. The DSS Tools can be used to develop a Specific DSS application directly as shown on the left half of the diagram. This is the same approach used to develop most traditional applications with tools such as a general purpose language, data access software, subroutine packages, *etc.* The difficulty with this approach for developing DSS applications is the constant change and flexibility which characterize them. A DSS changes character not only in repsonse to changes in the environment, but to changes in the way managers want to approach the problem. Therefore, a serious complicating factor in the use of basic tools is the need to involve the user directly in the change and modification of the Specific DSS.

APL was heavily used in the development of Specific DSS because it proved to be cheap and easy for APL programmers, especially the APL enthusiasts, to produce "throw-away" code which could be easily revised or discarded as the nature of the application changed. However, except for the few users who became members of the APL fan club, that language *did not* help capture the involvement of users in the building and modification of the DSS. The development and use of DSS Generators promises to create a "platform" or staging area from which Specific DSS can be constantly developed and modified with the cooperation of the user, and without heavy consumption of time and effort.

Evolving roles in DSS

All three levels of technology will probably be used over time in the development and operation of a DSS. Some interesting developments are occurring, however, in the roles that managers and technicians will play.

Figure 4 repeats part of the earlier diagram with a spectrum of five roles spread across the three levels.

- The *manager or user* is the person faced with the problem *or* decision — the one that must take action and be responsible for the consequences.

*The *intermediary* is the person who helps the user, perhaps merely as a clerical assistant to push the buttons of

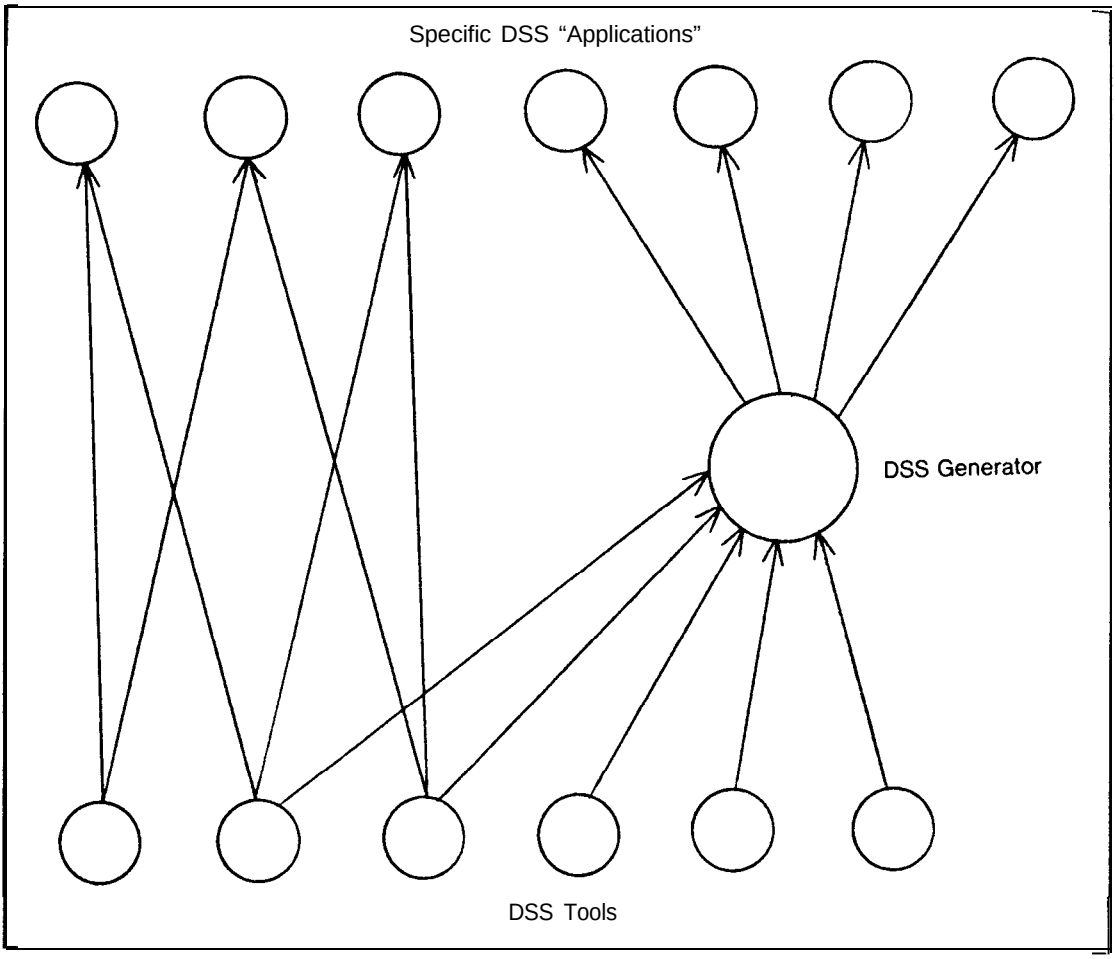

Figure 3. Three Levels of DSS Technology

the terminal, or perhaps as a more substantial "staff assistant" to interact and make suggestions.

- The DSS *builder* or facilitator assembles the necessary capabilities from the DSS Generator to "configure" the specific DSS with which the user/intermediary interacts directly. This person must have some familiarity with the problem area and also be comfortable with the information system technology components and capabilities.

*The *technical supporter* develops additional information system capabilities or components when they are needed as part of the Generator. New databases, new analysis models, and additional data display formats will be developed by the person filling this role. It requires a strong familiarity with technology, and a minor acquaintance with the problem or application area.

*The *too/smith* develops new technology, new languages, new hardware and software, improves the efficiency of linkages between subsystems, etc.

Two observations about this spectrum of roles are appropriate. First, it is clear that they do not necessarily align with individuals on a one-to-one basis. One person may assume several roles, or more than one person may be required to fill a

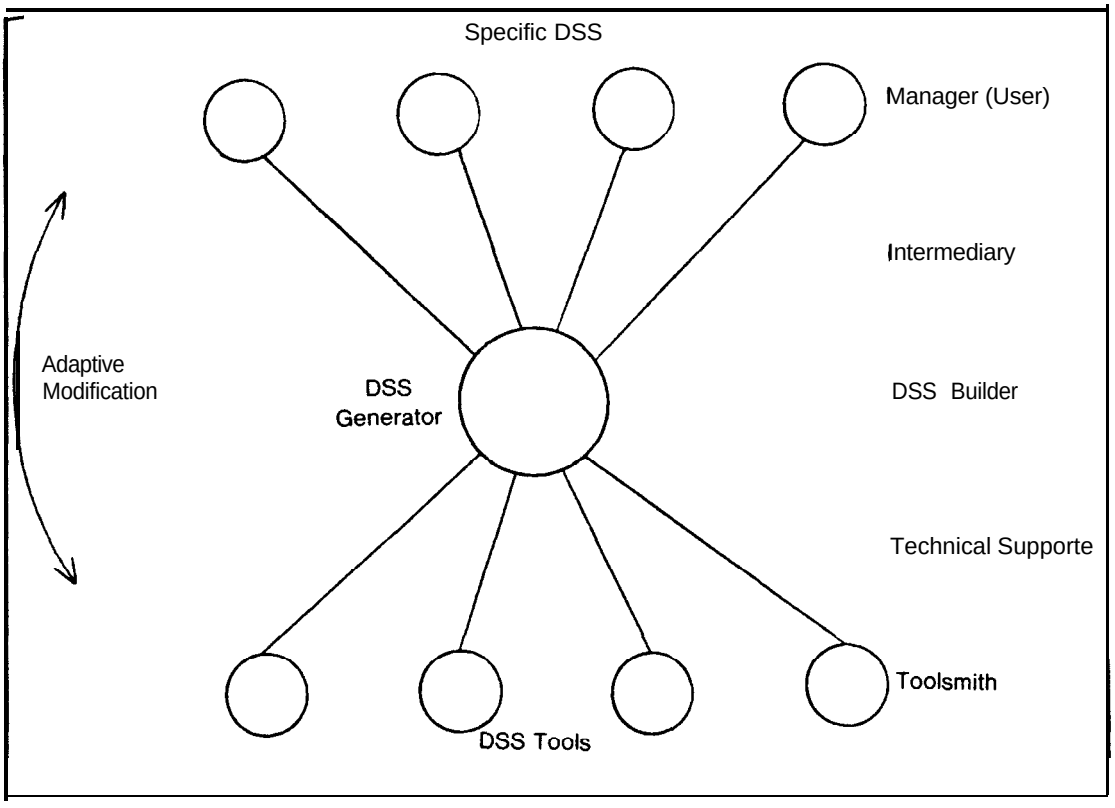

Figure 4. Three levels of DSS with Five Associated Roles **for Managers and Technicians**

role, The appropriate role assignment will generally depend on:

- the nature of the problem, particularly how narrow or broad;
- the nature of the person, particularly how comfortable the individual is with the computer equipment, language, and concepts; and
- the strength of the technology, particularly how user oriented it is.

Some managers do not need or want an intermediary, There are even a few chief executives who take the terminal home on weekends to write programs, thereby assuming the upper three or four roles. In fact, a recent survey of the users of IFPS shows that more than one third of them are middle and top level managers [45]. Decisions which require group consensus or systems design (builder) teams are examples of multiple persons per role.

Secondly, these roles appear similar to those present in traditional systems development, but there are subtle differences. The top two are familiar even in name for the development of many interactive or *online* systems. It is common practice in some systems to combine them into one "virtual" user for convenience. The user of the DSS, however, will play a much more active and controlling role in the design and development of the system than has been true in the past. The builder/technical supporter dichotomy is relatively close to the information specialist/system designer dichotomy discussed in the ACM curriculum recommendations [3]. Increasingly, however, the DSS builder resides in the functional area and not in the MIS department. The toolsmith is similar to a systems programmer, software designer, or computer scientist, but is increasingly employed by a hardware or software vendor, and not by the user's organization. The net result is less direct involvement in the DSS process by the information systems professional in the EDP/MIS department. (Some implications of

this trend are discussed later.) Moreover, the interplay between these roles is evolving into a unique development approach for a DSS.

The development approach for DSS

The very nature of a DSS requires a different design technique from traditional batch, or online, transaction processing systems. The traditional approaches for analysis and design have proven inadequate because there is no single comprehensive theory of decision making, and because of the rapidity of change in the conditions which decision makers face. Designers literally "cannot get to first base" because no one, least of all the decision maker or user, can define in advance what the functional requirements of the system should be. A DSS needs to be built with short, rapid feedback from users to ensure that development is proceeding correctly. It must be developed to permit change quickly and easily.

iterative Design

The result is that the most important four steps in the typical systems development process—analysis, design, construction, implementation—are combined into a single step which is iteratively repeated. Several names are evolving to describe this process including breadboarding [31], L'Approache Evolutive [14], and "middle out" [30]. The essence of the approach is that the manager and builder agree on a small but significant subproblem, then design and develop an initial system to support the decision making which it requires. After a short period of use, for instance, a few weeks, the system is evaluated, modified, and incrementally expanded. This cycle is repeated three to six times over the course of a few months until a relatively stable system is evolved which supports decision making for a cluster of tasks. The word "relatively" is important, because although the frequency and extent of change will decrease, it will never be stable. The system will always be changing, not as a necessary evil in response to imposed environmental changes, but as a conscious strategy on the part of the user and builder.

In terms of the three level model presented earlier, this process can be viewed as the iterative cycling between the DSS Generator and the Specific DSS as shown in Figure 4. With each cycle, capabilities are added to, or deleted from, the Specific DSS from those available in the DSS Generator. Keen depicts the expansion and growth of the system in terms of adding verbs which represent actions managers require [24]. Carlson adds more dimension by focusing on representations, operations, control, and memories as the elements of expansion and modification [11]. In another paper, Keen deals substantively with the interaction between the user, the builder, and the technology in this iterative, adaptive design process [25].

Note that this approach requires an unusual level of management involvement or management participation in the design. The manager is actually the iterative designer of the system; the systems analyst is merely the catalyst between the manager and the system, implementing the required changes and modifications.

Note also that this is different from the concept of "prototyping"; the initial system is real, live, and usable, not just a pilot test. The iterative process does not *merely* lead to a good understanding of the systems performance requirements, which are then frozen. The iterative changeability is actually built *into* the DSS as it is used over time. In fact, the development approach *becomes the system.* Rather than developing a system which is then "run" as a traditional EDP system, the DSS development approach results in the installation of an adaptive process in which a decision maker and a set of information system "capabilities" interact to confront problems while responding to changes from a variety of sources.

The Adaptive System

In the broad sense, the DSS is an adaptive system which consists of all three levels of technology in place and operating with the participants (roles), and the technology adapting to changes over time. Thus, the development of a DSS is actually the development and installation of this adaptive system. Simon describes such a system as one that adapts to changes of several kinds over three time horizons [34]. In the short run, the system allows a ***seach*** for answers within a relatively narrow scope. In the intermediate time horizon, the system *learns* by modifying its capabilities and activities, *i.e.,* the scope or domain changes. In the long run, the system

evolves to accommodate much different behavior styles and capabilities.

The three level model of a DSS is analogous to Simon's adaptive system. The Specific DSS gives the manager the capabilities and flexibility to search, explore, and experiment with the problem area, within certain boundaries. Over time, as changes occur in a task, the environment, and the user's behavior, the Specific DSS must learn to accommodate these changes through the reconfiguration of the elements in the DSS generator, with the aid of the DSS builder. Over a longer period of time, the basic tools evolve to provide the technology for changing the capabilities of the Generators out of which the Specific DSS is constructed, through the efforts of the toolsmith.

The ideas expressed above are not particularly new. Rapid feedback between the systems analyst and the client has been pursued for years. In the long run, most computer systems are adaptive systems. They are changed and modified during the normal system life cycle, and they evolve through major enhancements and extensions as the life cycle is repeated. However, when the length of that life cycle is shortened from three to five years to three to five months, or even weeks, there are significant implications. The resulting changes in the development approach and the traditional view of the systems life cycle promises to be one of the important impacts of the growing use of a DSS.

Performance Objectives and Capabilities

Most of the foregoing discussion has dealt with some aspects of the technological and organizational contexts within which a DSS will be built and operated. The second part of the framework deals with what a DSS must accomplish, and what capabilities or characteristics it must have. The three levels of hardware/software technology and the corresponding three major "stakeholders" or interested parties in the development and use of a DSS can be used to identify the characteristics and attributes of a DSS.

At the top level are the *managers or users* who are primarily concerned with what the Specific DSS can do for them. Their focus is the problem solving or decision making task they face, and the organizational environment in which they operate. They will assess a DSS in terms of the assistance they receive in pursuing these tasks. At the level of the DSS Generator, the *builders* or designers must use the capabilities of the generator to configure a Specific DSS to meet the manager's needs. They will be concerned with the capabilities the Generator offers, and how these capabilities can be assembled to create the specific DSS. At the DSS tool level, the *"too/smiths"* are concerned with the development of basic technology components, and how they can be integrated to form a DSS Generator which has the necessary capabilities.

The attributes and characteristics of a DSS as viewed from each level must be examined. From the manager's view, six general performance objectives for the Specific DSS can be identified. They are not the only six that could be identified, but as a group they represent the overall performance of a DSS that seems to be expected and desirable from a managerial viewpoint. The characteristics of the DSS Generator from the viewpoint of the builder are described by a conceptual model which identifies performance characteristics in three categories: dialogue handling or the man-machine interface, database and database management capability, and modeling and analytic capability. The same three part model is used to depict the viewpoint of the "toolsmith," but from the aspect of the technology, tactics, and architecture required to produce those capabilities required by the builders.

Manager's view: performance objectives

The following performance requirements are phrased using the normative word "should." It is likely that no Specific DSS will be required to satisfy all six of the performance requirements given here. In fact, it is important to recall that the performance criteria for any Specific DSS will depend entirely on the task, the organizational environment, and the decision maker(s) involved. Nevertheless, the following objectives collectively represent a set of capabilities which characterize the full value of the DSS concept

from the manager/user point of view. The first three pertain to the type of decision making task which managers and professionals face. The latter three relate to the type of support which is needed.

A DSS should provide support for decision making, but with emphasis on semi-structured and unstructured decisions. These are the types of decisions that have had little or no support from EDP, MIS, or management science/operations research (MS/OR) in the past. It might be better to refer to "hard" or underspecified problems, because the concept of "structure" in decision making is heavily dependent on the cognitive style and approach to problem solving of the decision maker. It is clear from their expressed concerns however, that managers need additional support for certain kinds of problems.

2 *A DSS should provide decision making support for managers at* all levels, *assisting* in *integration between the levels whenever appropriate.* This requirement evolves from the realization that managers at *all* organizational levels face "tough" problems as described in the first objective above. Moreover, a major need articulated by managers, is the integration and coordination of decision making by several managers dealing with related parts of a larger problem.

3. *A DSS should support decisions which are interdependent as well as those that are independent.* Much of the early DSS work inferred that a decision maker would sit at a terminal, use a system, and develop a decision *alone.* DSS development experience has shown that a DSS must accommodate decisions which are made by groups or made in part by several people in sequence. Keen and Hackathorn [24] explore three decision types as:

**Independent.* A decision maker has full responsibility and authority to make a complete implementable decision.

**Sequential Interdependent.* A decision maker makes part of a decision which is passed on to someone else.

- *Pooled Interdependent.* The decision must result from negotiation and interaction among decision makers.

Different capabilities will be required to support each type of decision—personal support, organizational support, and group support respectively.

4. *A DSS should support all phases of the decision making process.* A popular model of the decision making process is given in the work of Herbert Simon [33]. He characterized three main steps in the process as follows:

- *Intelligence.* Searching the environment for conditions calling for decisions. Raw data is obtained, processed, and examined for clues that may identify problems.

- *Design.* Inventing, developing, and analyzing possible courses of action. This involves processes to understand the problem, generate solutions, and test solutions for feasibility.

**Choice.* Selecting a particular course of action from those available. A choice is made and implemented.

Although the third phase includes implementation, many authors feel that it is significant enough to be shown separately. It has been added to Figure 5 to show the relationships between the steps. Simon's model also illustrates the contribution of MIS/EDP and MS/OR to decision making. From the definition of the three stages given above, it is clear that EDP and MIS, in the narrow sense, have made major contributions to the intelligence phase, while MS/OR has been primarily useful at the choice phase. There has been no substantial support for the design

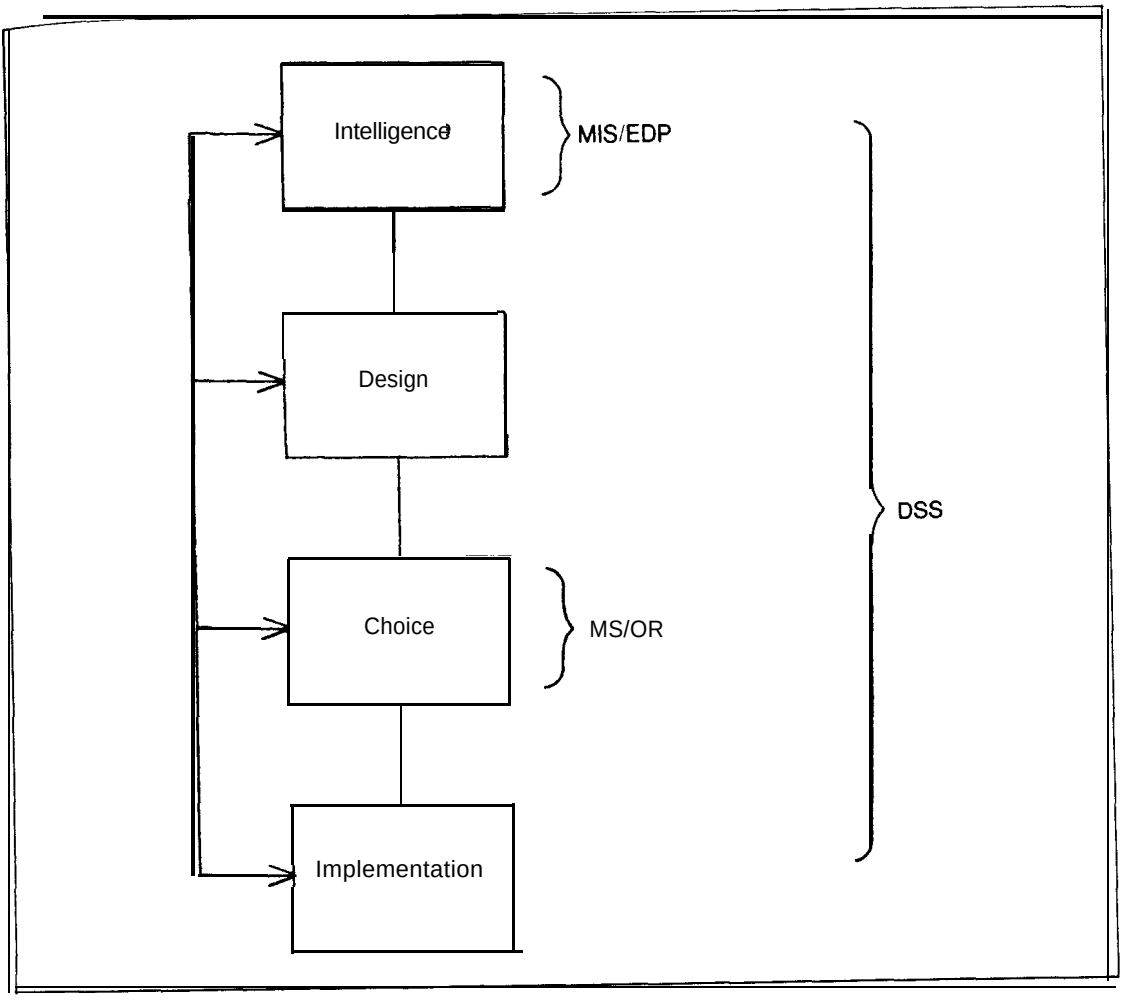

Figure 5. **Phases of Decision Making**

phase, which seems to be one of the primary potential contributions of a DSS. There also has been very little support from traditional systems for the implementation phase, but some early experience has shown that a DSS can make a major contribution here also [42].

5. *A DSS should support a variety of decision making processes, but not be dependent on any one.* Simon's model, though widely accepted, is only one model of how decisions are actually made. In fact, there is no universally accepted model of the decision making process, and there is no promise of such a general theory in the foreseeable future. There are too many variables, too many different types of decisions, and too much variety in the characteristics of decision makers. Consequently, a very important characteristic of a DSS is that it provide the decision maker with a set of capabilities to apply in a sequence and form that fits each individual cognitive style. In short, a DSS should be process independent, and user driven or controlled.

6 *Finally, a DSS should be easy to use.* A variety of terms have been used to describe this characteristic including flexibility, user friendly, nonthreatening, etc. The importance of this characteristic is underscored by the discretionary latitude of a DSS's clientele. Although some systems which require heavy organizational support or group support may limit the discretion somewhat, the user of a DSS has much more latitude to ignore or circumvent the system than the user of a more traditional transaction system or required reporting system. Therefore, a DSS must "earn" its users' allegiance by being valuable and convenient.

The builder's view: technical capabilities

The DSS Builder has the responsibility of drawing on computer based toots and techniques to provide the decision support required by the manager. DSS Tools can be used directly, but it is generally more efficient and effective to use a DSS Generator for this task. The Generator must have a set of capabilities which facilitate the quick and easy configuration of a Specific DSS and modification in response to changes in the manager's requirements, environment, tasks, and thinking approaches. A conceptual model can be used to organize these capabilities, both for the builders and for the "toolsmith" who will develop the technology to provide these capabilities.

The old "black box" approach is helpful here, starting with the view of the system as a black box, successively "opening" the boxes to understand the subsystems and how they are interconnected. Although the DSS is treated as the black box here, it is important to recall that the overall system is the decision *making* system, consisting of a manager/user who uses a DSS to confront a task in an organizational environment.

Opening the large DSS box reveals a database, a model base, and a complex software system for linking the user to each of them as shown in Figure 6. Opening each of these boxes reveals that the database and model base have some interrelated components, and that the software system is comprised of three sets of capabilities: database management software (DBMS), model base management software (MBMS), and the software for managing the interface between the user and the system, which might be called the dialogue generation and management software (DGMS). These three major subsystems provide a convenient scheme for identifying the technical capability which a DSS must have. The key aspects in each category that are critical to a DSS from the Builder's point of view, and a list of capabilities which will be required in each category must now be considered.

The data subsystem

The data subsystem is thought to be a well understood set of capabilities because of the rapidly maturing technology related to databases and their management. The typical advantages of the database approach, and the powerful functions of the DBMS, are also important to the development and use of a DSS. There are, however, some significant differences between the Database/Data Communication approach for traditional systems, and those applicable for a DSS. Opening the Database box summarizes these key characteristics as shown in Figure 7.

First is the importance of a much richer set of data sources than are usually found in typical non-DSS applications. Data must come from external as well as internal sources, since decision making, especially in the upper management levels, is heavily dependent on external data sources. In addition, the typical accounting oriented transaction data must be supplemented with non-transactional, non-accounting data, some of which has not been computerized in the past.

Another significant difference is the importance of the data capture and extraction process from this wider set of data sources. The nature of a DSS requires that the extraction process, and the DBMS which manages it, be flexible enough to allow rapid additions and changes in response to unanticipated user requests. Finally, most successful DSS's have found it necessary to create a DSS database which is logically separate from other operational databases. A partial set of capabilities required in the database area can be summarized by the following:

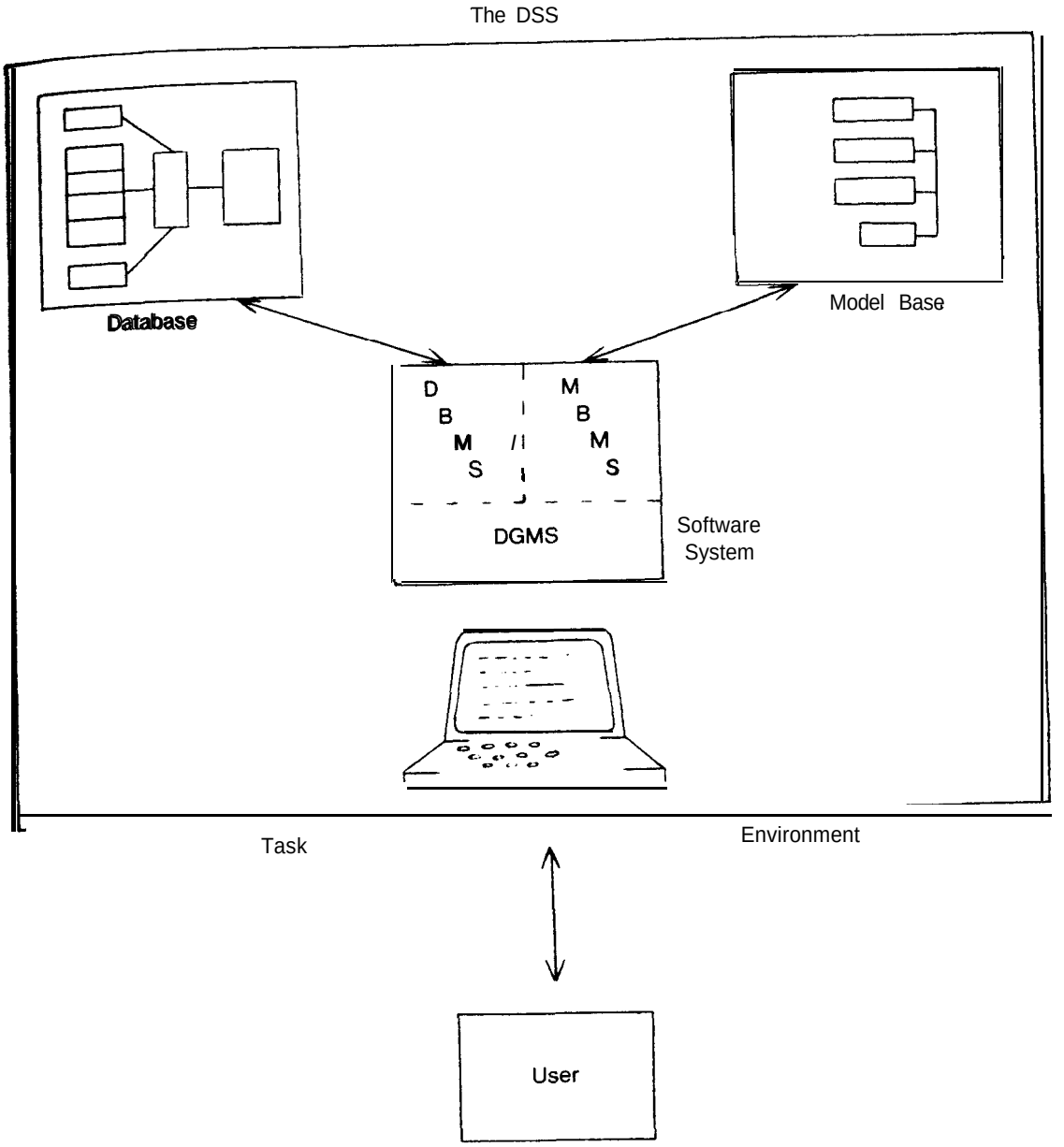

Figure 6. Components of the *DSS*

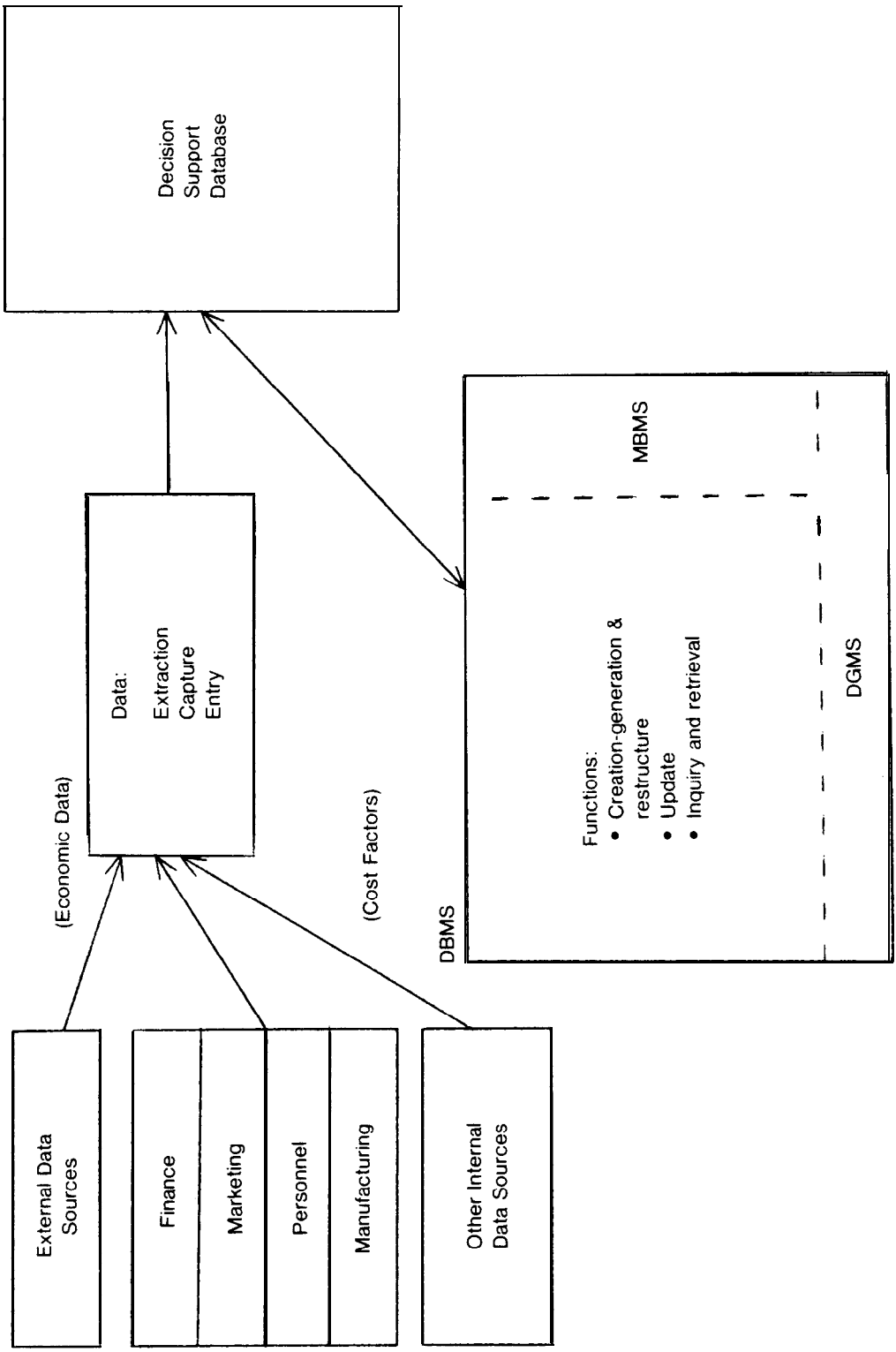

Figure 7. The Data Subsyst

- the ability to combine a variety Of data sources through a data capture and extraction process:
- the ability to add and delete data sources quickly and easily;
- the ability to Portray logical data structures in user terms so the user understands what is available and can specify needed additions and deletions;
- the ability to handle personal and unofficial data so the user can experiment with alternatives based on personal judgment; and
- the ability to manage this wide variety of data with a full range of data management functions.

The model subsystem

A very promising aspect of a DSS is its ability to integrate data access and decison models. It does so by imbedding the decision models in an information system which uses the database as the integration and communication mechanism between models. This characteristic unifies the strength of data retrieval and reporting from the EDP field and the significant developments in management science in a way the manager can use and trust.

The misuse and disuse of models have been widely discussed [21, 28, 36, 39]. One major problem has been that model builders were frequently preoccupied with the structure of the model. The existence of the correct input data and the proper delivery of the output to the user was assumed. In addition to these heroic assumptions, models tended to suffer from inadequacy because of the difficulty of developing an integrated model to handle a realistic set of interrelated decisions. The solution was a collection of separate models, each of which dealt with a distinct part of the problem. Communication between these related models was left to the decision maker as a manual and intellectual process.

A more enlightened view of models suggests that they be imbedded in an information system with the database as the integration and communication mechanism between them. Figure 8 summarizes the components of the model base "box." The model creation process must be flexible, with a strong modeling language and a set of building blocks, much like subroutines, which can be assembled to assist the modeling process. In fact, there are a set of model management functions, very much analogous to data management functions. The key capabilities for a DSS in the model subsystems include:

- the ability to create new models quickly and easily;
- the ability to catalog and maintain a wide range of models, supporting all levels of management;
- the ability to interrelate these models with appropriate linkages through the database:
- the ability to access and integrate model "building blocks;" and
- the ability to manage the model base with management functions analogous to database management (e.g., mechanisms for storing, cataloging, linking, and accessing models).

For a more detailed discussion of the model base and its management see (37. 38, 46].

The User System Interface

Much of the power, flexibility, and usability characteristics of a DSS are derived from capabilities in the user system interface. Bennett identifies the user, terminal, and software system as the components of the interface subsystem [5]. He then divides the dialogue, or interface experience itself into three parts as shown in Figure 9:

1. *The action language* — what the user *can do* in communicating with the system. It includes such options as the availability of a regular keyboard, function keys, touch panels, joy stick, voice command, etc.
2. *The display or presentation language* — what the user sees. The display language includes options such as character or line printer, display

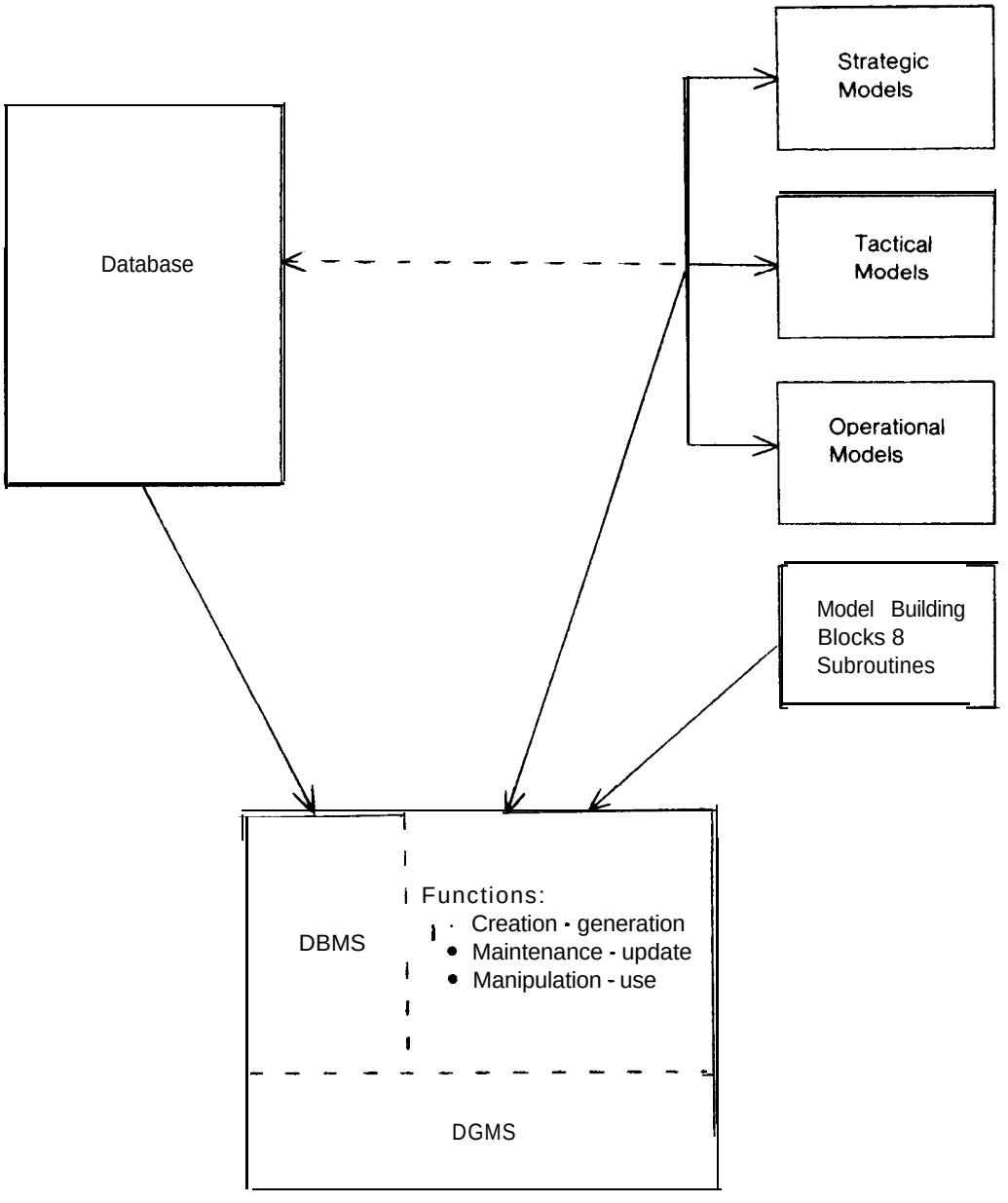

Figure 8. The Models Subsystem

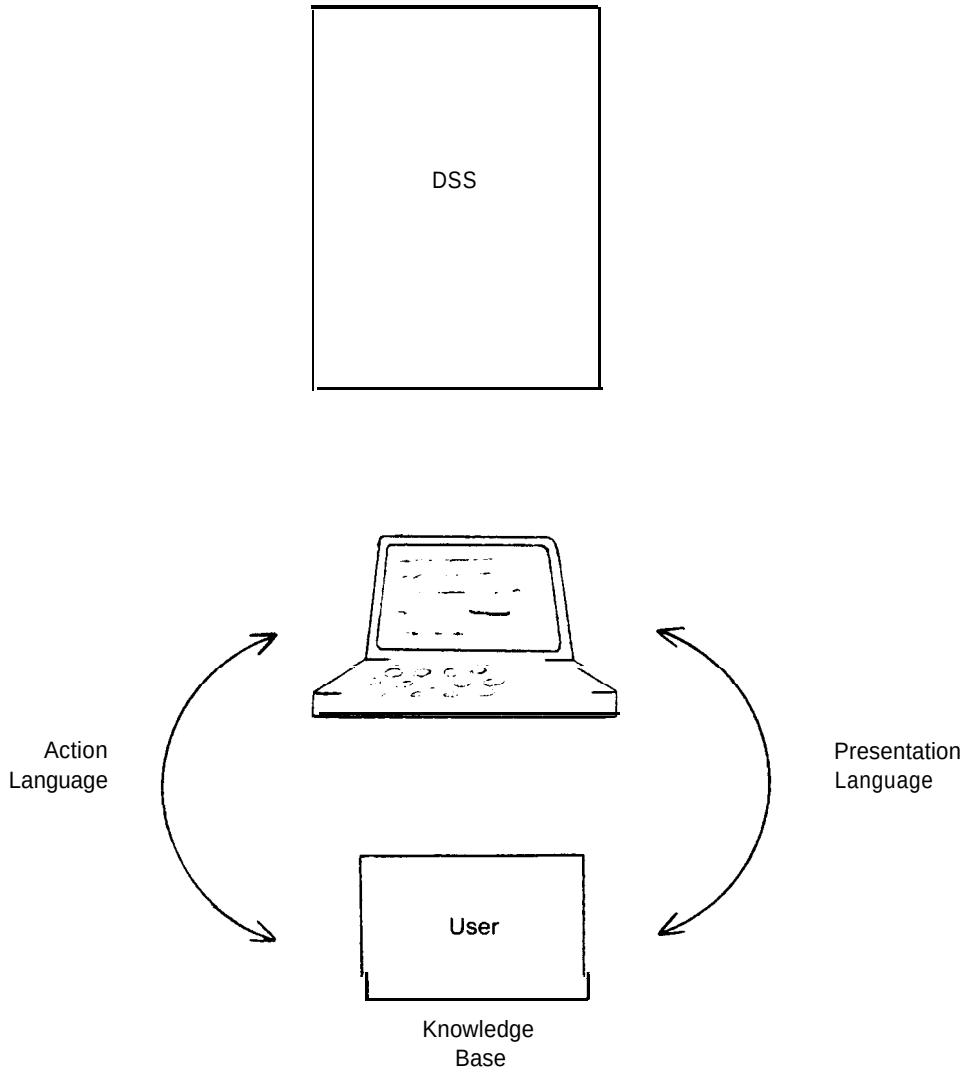

Figure 9. The User System Interface

screen, graphics, color, plotters, audio output, etc.

3. *The knowledge base* — what the user *must know.* The knowledge base consists of what the user needs to bring to the session with the system in order to effectively use it. The knowledge may be in the user's head, on a reference card or instruction sheet, in a user's manual, in a series of "help" commands available upon request, *etc.*

The "richness" of the interface will depend on the strength of capabilities in each of these areas.

Another dimension of the user system interface is the concept of "dialogue style." Examples include the questions/answer approach, command languages, menus, and "fill in the blanks." Each style has pro's and con's depending on the type of user, task, and decision situation. For a more detailed discussion of dialogue styles see [13].

Although this just scratches the surface in this important area, a partiil set of desirable capabilities for a DSS generator to support the user/system interface includes:

- the ability to handle a variety of dialogue styles, perhaps with the ability to shift among them at the user's choice;
- the ability to accommodate **user actions** in a variety of media;
- the ability to present data in a variety of formats and media; and
- the ability to provide flexible support for the users' knowledge base.

The too/smith view: the underlying technology

The toolsmith is concerned with the science involved in creating the information technology to support a DSS, and the architecture of combining the basic tools into a coherent system. The same three part model can be used to describe the toolsmith's concerns because the tools must be designed and combined to provide the three sets of capabilities.

Each of the three areas-dialogue, data handling, and model handling-has received a fair amount of attention from toolsmiths in the past. The topic of DSS and the requirements it imposes has put these efforts in a new perspective revealing how they can be interrelated to increase their collective effectiveness. Moreover, the DSS requirements have revealed some missing elements in existing efforts, indicating valuable potential areas for development.

Dialogue **Management**

There has been a great deal of theoretical and some empirical work on systems requirements for good man/machine interface. Many of these studies are based on watching users' behavior in using terminals, or surveying users or programmers to ascertain what they want in interactive systems [10, 16]. A recent study examines a series of interactive applications, many of which are DSS's, to assess the *type* of software capabilities required by the applications [43]. This study led directly to some creative work on the software architecture for dialogue generation and management systems (DGMS) as characterized in the model of the previous section [12]. This research uses a relation as the data structure for storing each picture or "frame" used in the system, and a decision table for storing the control mechanism for representing the potential users' option in branching from one frame to another.

Data Management

Most of the significant work in the database management area during the past several years is aimed at transaction processing against large databases. Large DBMS's generally have inquiry/retrieval and flexible report preparation capabilities, but their largest contribution has been in the reduction of program maintenance costs through the separation of application programs and data definitions. On the other hand, DBMS work has generally had a rather naive view of the user and the user's requirements. A DSS user will not be satisfied merely with the capability to issue a set of retrieval commands which select items from the database, or even to display those selected items in a report with the flexible definition of format and headings. A DSS user needs to interact repeatedly and creatively with a relatively

small set of data. The user may only need 40-l 00 data variables, but they must be the *right ones;* and what is right may change from day to day and week to week. Required data will probably include time series data which are not handled comprehensively by typical DBMS's Better ways are needed to handle and coordinate time series data as well as mechanisms for capturing, processing, and tagging judgmental and probabilistic data. Better ways are also needed for extracting data from existing files and capturing data from previously non-computerized sources. The critical area of data extraction with fast response, which allows additions and deletions to the DSS database from the large transaction database was a major contribution of the GADS work [8,29]. In short, the significant development in database technology needs to be focused and extended in some key areas in order to directly serve the needs of a DSS.

Model **Management**

The area of model creation and handling may have the greatest potential contribution to a DSS. So far, the analytic capability provided by systems has evolved from statistical or financial analysis subroutines which can be called from a common command language. More recently, modeling languages provide a way of formulating interrelationships between variables in a way that permits the creation of simulation or "what if" models. As we noted earlier, many of the currently viable DSS Generators have evolved from these efforts. Early forms of "model management" seem to be evolving from enhancements to some modeling languages, which permit a model of this type to be used for sensitivity testing or goal seeking by specifying target and flexibility variables.

The model management area also has the potential for bringing some of the contributions of artificial intelligence (AI) to bear on a DSS. MYCIN, a system to support medical diagnosis, is based on "production rules," in the AI sense, which play the role of models in performing analytic and decision guidance functions [15]. A more general characterization of "knowledge management" as a way of handling models and data has also been tentatively explored [7]. More recent work proposes the use of a version of semantic networks for model representation [17]. Though this latter work is promising, AI research has shown the semantic network approach to be relatively inefficient with today's technology. Usable capabilities in model management in the near future are more likely to evolve from modeling languages, expanded subroutine approaches, and in some cases, AI production rules.

Issues for the Future

At this stage in the development of the DSS area, issues, problems, and fruitful directions for further research/development are plentiful. At a "task force" meeting this summer, thirty researchers from twelve countries gathered to discuss the nature of DSS's and to identify issues for the future. Their list, developed in group discussions over several days, was quite long [19]. The issues given here, phrased as difficult questions, seem to be the ones that must be dealt with quickly, lest the promise and potential benefits of DSS's be diluted or seriously delayed.

What's a DSS?

Earlier it was noted that some skeptics regard DSS as "just another buzz word." This article has shown that there is a significant amount of content behind the label. The danger remains, however, that the bandwagon effect will outrun our ability to define and develop potential contributions of a DSS. The market imperatives of the multi-billion dollar information systems industry tend to generate pressures to create simple labels for intuitively good ideas. It happened in many cases, but not all, of course, with MIS. Some companies are still trying to live down the aftereffects of the overpromise/under-undelivery/ disenchantment sequence from the MIS bandwagon of the late '60's. Eventually, a set of minimal capabilities or characteristics which characterize a DSS should evolve. In the short range, a partial solution is education — supplying managers with intellectual ammunition they can use in dealing with vendors. Managers should and must ask sharp, critical questions about the capabilities of any purported DSS, matching them against what is really needed.

What is really needed?

After nearly two decades of advancements in information technology, the real needs of

managers from an information system are not well understood. The issue is further complicated by the realization that managers' needs and the needs of other "knowledge workers" with which they interact, are heavily interdependent. The DSS philosophy and approach has already shed some light on this issue by emphasizing "capabilities" — the ability for a manager to do things with an information system — rather than just "information needs" which too often infer data items and totals on a report.

Nevertheless, it is tempting to call for a hesitation in the development of DSS's until decision making and related managerial activities are fully understood. Though logically appealing, such a strategy is not practical. Neither the managers who face increasingly complex tasks, nor the information systems industry which has increasingly strong technology to offer, will be denied. They point out that a truly comprehensive theory of decision making has been pursued for years with minimum success.

A potential resolution of this problem is to develop and *use* a DSS in a way that reveals what managers can and should receive from an information system. For example, one of Scott Morton's early suggestions was that the system be designed to capture and track the steps taken by managers in the process of making key decisions, both as an aid to the analysis of the process, and as a potential training device for new managers.

The counterpart of the "needs" issue is the extent to which the system meets those needs, and the value of the performance increase that results. Evaluation of a DSS will be just as difficult, and important, as the evaluation of MIS has been. The direct and constant involvement of users, the ones in the best position to evaluate the systems, provides a glimmer of hope on this tough problem. Pursuit of these two tasks together may yield progress on both fronts with the kind of synergistic effect often sought from systems efforts. The iterative design approach and the three levels of technology afford the opportunity, if such a strategy is developed from the beginning.

Who will do it?

A series of organizational issues will revolve around the roles and organizational placement of the people who will take the principle responsibility for the development of DSS's. Initiative and guidance for DSS development efforts frequently come from the user area, not from the EDP/MIS area. Yet current technology still requires technical support from the information systems professional. The DSS builder may work for the vice president of finance, but the technical support role is still played by someone in the MIS department. To some extent, the demand for a DSS supports the more general trend to distribute systems development efforts out of the MIS department into the user department. The difference is that many DSS software systems, or generators, specifically attempt to directly reach the end user without involvement of the MIS group. The enlightened MIS administrator considers this a healthy trend, and willingly supplies the required technical support and coordination. Less enlightened DP administrators often see it as a threat. Some companies have set up a group specifically charged with developing DSS type applications. This strategy creates a team of "DSS Builders" who can develop the necessary skills in dealing with users, become familiar with the available technology, and define the steps in the developmental approach for DSS's.

How *should it be done?*

One of the pillars on which the success of DSS rests, is the iterative development or adaptive design approach. The traditional five to seven stage system development process and the system life cycle concept have been the backbone of systems analysis for years. Most project management systems and approaches are based on it. The adaptive design approach, because it combines all the stages into one quick step which is repeated, will require a redefinition of system development milestones and a major modification of project management mechanisms. Since many traditional systems will not be susceptible to the iterative approach, a way is also needed for deciding when an application should be developed in the new way instead of the traditional way. The outline of the approach described earlier is conceptionally straightforward

for applications that require only personal support. It becomes more complicated for group or organizational support when there are multiple users. In short, DSS **builders will** need to develop a set of milestones, checkpoints, documentation strategies, and project management procedures for DSS applications, and recognize when they should be used.

How much can be done?

The final issue is a caveat dealing with the limitations of technical solutions to the complexity faced by managers and decision makers. As information systems professionals, we must be careful not to feel, or even allow others to feel, that we can develop or devise a technological solution to all the problems of management. Managers will always "deal with complexity in a state of perplexity" — it is the nature of the job. Information technology can, and is, making a major contribution to improving the effectiveness of people in this situation, but the solution will never be total. With traditional systems, we continually narrow the scope and definition of the system until we know it will do the job it is required to do. If the specification/design/construction/implementation process is done right, the system is a success, measured against its original objectives. With a DSS. the user and his systems capabilities are constantly pursuing the problem, but the underspecified nature of the problem insures that there will never be a complete solution. Systems analysts have always had a little trouble with humility, but the DSS process requires a healthy dose of modesty with respect to the ability of technology to solve all the problems of managers in organizations.

Conclusion

The "Framework for Development" described above attempts to show the dimensions and scope of DSS in a way that will promote the further development of this highly promising type of information system.

1. The relationships between EDP. MIS, and DSS show that DSS is only one of several important technology systems for improving organizational performance, and that DSS development efforts must carefully integrate with these other systems.
2. The three levels of technology and the interrelationships between people that use them provide a context for organizing the development effort.
3. The iterative design approach shows that the ultimate goal of the DSS development effort is the installation of an adaptive system consisting of all three levels of technology and their users operating and adapting to *changes* over time.
4. The performance objectives show the types of decision making to be served by, and the types of support which should be built into, a DSS as it is developed.
5. The three technical capabilities illustrate that development efforts must provide the DSS with capabilities in dialogue management, data management, and model management.
6. The issues discussed at the end of the article identify some potential roadblocks that must be recognized and confronted to permit the continued development of DSS.

In closing, it should now be clear that DSS is more than just a "buzz word," but caution must be used in announcing a new "era" in information systems. Perhaps the best term is a "DSS Movement" as user organizations, information systems vendors, and researchers become aware of the field, its potential, and the many unanswered questions. Events and mechanisms in the DSS Movement include systems development experience in organizations, hardware/software developments by vendors, publishing activities to report experience and research, and conferences to provide a forum for the exchange of ideas among interested parties.

It is clear that the momentum of the DSS Movement is building. With appropriate care and reasonable restraint, the coordinated efforts of managers, builders, toolsmiths, and researchers

can converge in the development of a significant set of information systems to help improve the effectiveness of organizations and the people who work in them.

References

[1] Alter, S. "A Taxonomy of Decision Support Systems," *Sloan Management Review,* Volume 19, Number 1, Fall 1977, pp. 39-56.

[2] Alter, S. *Decision Support Systems: Current Practice and Continuing Challenges,* Addison-Wesley Publishing Co., Reading, Massachusetts, 1980.

[3] Ashenhurst, R. L. "Curriculum Recommendations for Graduate Professional Programs in Information Systems," *ACM Communications,* Volume 15, Number 5, May 1972, pp. 363-398.

[4] Barbosa, L. C. and Hirko, R. G. "Integration of Algorithmic Aids into Decision Support Systems," MIS *Quarterly,* Volume 4, Number 1, March 1980, pp. l-l 2.

[5] Bennett, J. "User-Oriented Graphics, Systems for Decision Support in Unstructured Tasks," in *User-Oriented* **Design of** *Interactive Graphics Systems,* in S. Treu (ed.), Association for Computing Machinery, New York, New York, 1977, pp. 3-11.

[6] Boeing Computer Services, c/o Mr. Park Thoreson, P. 0. Box 24346, Seattle, Washington, 98124.

[7] Bonezek, H., Hosapple, C. W., and Whinston. A. "Evolving Roles of Models in Decision Support Systems," *Decision Sciences,* Volume 11, Number 2, April 1980, pp. 337-356.

[8] Carlson, E. D., Bennett, J., Giddings, G., and Mantey, P. "The Design and Evaluation of an Interactive Geo-Data Analysis and Display System," *Information Processing- 74,* North Holland Publishing Co., Amsterdam, Holland, 1974.

[9] Carlson, E. D., and Sutton, J. A. "A Case Study of Non-Programmer Interactive Problem Solving, *IBM Research Report RJ1382,* San Jose, California, 1974.

[10] Carlson, E. D., Grace, B. F. and Sutton, J. A. "Case Studies of End User Requirements for Interactive Problem-Solving Systems," MIS *Quarter/y,* Volume 1, Number 1, March 1977, pp. 51-63.

[11] Carlson, E. D. "An Approach for Designing Decision Support Systems," *Proceedings,* 1 1 th Hawaii International Conference on Systems Sciences, Western Periodicals Co., North Hollywood, California, 1978, pp. 76-96

[12] Carlson, E. D. and Metz, W. "Integrating Dialog Management and Data Management," IBM *Research Report RJ2738,* February 1, 1980, San Jose, California.

[13] Carlson, E. D. "The User-Interface for Decision Support Systems," unpublished working paper, IBM Research Laboratory, San Jose, California.

[14] Courbon, J., Drageof, J., and Jose, T. "L'Approache Evolutive," *Information Et Gestion No. 1 03,* Institute d' Administration des Enterprises, Grenoble, France, January-February 1979, pp. 51-59.

[15] Davis, R. "A DSS for Diagnosis and Therapy," *Data Base,* Volume 8, Number 3, Winter 1977, pp. 58-72.

[16] Dzida, W., Herda, S., and Itzfeldt, W. D. "User-Perceived Quality of Software Interactive Systems," *Proceedings,* Third Annual Conference on Engineering (IEEE) Computer Society, Long Beach, California, 1978, pp. 188-l 95.

[17] Elam, J., Henderson, J., and Miller, L. "Model Management Systems: An Approach to Decision Support in Complex Organizations," *Proceedings,* Conference on Information Systems, The Society for Management Information Systems, Philadelphia, Pennsylvania, December 1980.

[18] Execucom Systems Corporation, P. 0. Box 9758, Austin, Texas, 78766.

[19] Fick, G. and Sprague, R. H., Jr., (eds.). *Decision Support Systems: Issues and Challenges,* Pergamon Press, Oxford, England, forthcoming in 1981.

[20] Gerrity, T. P., Jr. "Design of Man-Machine Decision Systems: An Application to Portfolio Management," *Sloan Management Review 72,* Volume 12, Number 2, Winter 1971, pp. 59-75.

[21] Hayes, R. H. and Noland, R. L. "What Kind of Corporate Modeling Functions Best?" *Harvard Business Review,* Volume 52,

May-June 1974, pp. 102-l 12.

[22] Head, R. "Management Information Systems: A Critical Appraisal," *Datamation,* Volume 13, Number 5, May 1967, PP. 22-28.

[23] Keen, P. G. W. and Scott Morton, M. S. *Decision Support Systems: An Organizational Perspective,* Addison-Wesley Publishing Company, Reading Massachusetts, 1978.

[24] Keen, P. G. W. and Hackathorn. R. D. "Decision Support Systems and Personal Computing," Department of Decision Sciences, The Wharton School, The University of Pennsylvania, Working Paper 79-01-03, Philadelphia, Pennsylvania, April 3, 1979.

[25] Keen, P. G. W. "Adaptive Design for DSS," *Database,* Volume 12, Numbers 1 and 2, Fall 1980, pp. 15-25.

[26] Keen, P'. G. W. "Decision Support Systems: A Research Perspective," in *Decision Support Systems: Issues and Challenges,* Pergamon Press, Oxford, England, 1981.

[27] Kroeber, H. W., Watson, H. J., and Sprague, R. H., Jr. "An Empirical Investigation and Analysis of the Current State of Information Systems Evolution," *Journal of Information and Management,* Volume 3, Number 1, February 1980, pp. 35-43.

[28] Little, J. D. C. "Models and Managers: The Concept of a Decision Calculus," *Management Science,* Volume 16, Number 8, April 1970, pp. B466-485.

[29] Mantey, P. E. and Carlson, E. D. "Integrated Geographic Data Bases: The GADS Experience," IBM Research Division, IBM *Research Report* **RJ2702,** San Jose, California, December 3, 1979.

[30] Ness, D. N. "Decision Support Systems: Theories of Design," presented at the Wharton Office of Naval Research Conference on Decision Support Systems, Philadelphia, Pennsylvania, November 4-7, 1975.

[31] Scott, J. H. "The Management Science Opportunity: A Systems Development Management Viewpoint," MIS *Quarterly,* Volume 2, Number 4, December 1978, pp. 59-61.

1321 Scott Morton, M. S. *Management Decision Systems: Computer Based Support for Decision Making,* Division of Research, Harvard University, Cambridge, Massachusetts, 197 1.

(331 Simon, H. *The New Science of Management Decision,* Harper and Row, New York, New York, 1960.

[34] Simon, H. "Cognitive Science: The Newest Science of the Artificial," *Cognitive Science,* Volume 4, 1980. pp. 33-46.

[35] Society for Management Information Systems, *Proceedings of the Eleventh Annual Conference,* Chicago, Illinois. September 1 O-l 3, 1979, pp. 45-56.

[36] Sprague, R. H. and Watson, H. J. "MIS Concepts Part I," *Journal of Systems Management,* Volume 26, Number 1, January 1975, pp. **34-37.**

[37] Sprague, R. H. and Watson, H. J. "Model Management in MIS," *Proceedings, 7th National AIDS,* Cincinnati, Ohio, November 5, 1975, pp. 213-215.

[38] Sprague, R. H. and Watson, H. "A Decision Support System for Banks," *Omega - The International Journal of Management Science,* Volume 4, Number 6, 1976, pp. 657-671.

[39] Sprague, R. H. and Watson, H. J. "Bit by Bit: Toward Decision Support Systems," *California Management Review,* Volume XXII, Number 1, Fall 1979, pp. 60-68.

[40] Sprague, R. H. "Decision Support Systems — Implications for the Systems Analysts," *Systems Analysis and Design: A Foundation for the 1980's,* Elsevier-North Holland, New York, New York, 1980, in press.

[41] Sprague, R. H. "A Framework for Research on Decision Support Systems," in *Decision Support Systems: Issues and Challenges,* Fick, G. and Sprague, R. H. (eds.), Pergamon Press, Oxford, England 1981, in press.

1421 Sutton, J. "Evaluation of a Decision Support System: A Case Study with the Office Products Division of IBM," San Jose, California: *IBM Research Report FJ2214* 1978.

[43] Sutton, J. A., and Sprague, R. H. "A Study of Display Generation and Management in Interactive Business Applications," *IBM Research Report No. RJ2392,* IBM Research Division, San Jose, California, November 9, 1978.

[44] **TYMSHARE. 20705** Valley Green Driver, Cupertino, California, 95014.

[45] Wagner, G. R. "DSS: Hypotheses and Inferences," Internal Report, EXECUCOM Systems Corporation, Austin, Texas, 1960.

[46] Will, Hart J. "Model Management Systems," in *Information Systems and Organizational Structure,* E Grochla and H. Szyperski (eds.), Walter de Gruyter, New York, New York, 1975, pp. 467-463.

About the Author

Dr. Ralph H. Sprague, *Jr. is Professor and Chairman of the Decision Sciences Department in the College of Business Administration at the University of Hawaii. His research and consulting specialty is computer based information systems for management, with particular emphasis on Decision Support Systems. He has lectured widely on this and related subjects including a series of seminars this past summer in Kuwait, Vienna, Austria, and Norway. In 19 79 he won a national award from The Society for Management Information Systems for his article describing the computer based financial planning system at Louisiana National Bank. He has published articles in* California Management Review, Decision Sciences, Management Accounting, Journal of Bank Research, *and* Journal of Systems Management.