Learning Objectives

◆ Understand the definition of a decision support system (DSS) based on three common themes: problem structure, decision outcome, and managerial control
◆ Understand the benefits and limitations of DSS use
◆ Be familiar with the history of DSSs
◆ Grasp the five basic components of a DSS
◆ Learn the roles of data and model management systems
◆ Understand the functions of a DSS knowledge base
◆ Learn the importance of the user interface in a DSS
◆ Learn the user roles and patterns of DSS use
◆ Gain an understanding of the categories and classes of DSSs that are essential in designing or implementing a new system
Faced with increasing competition from dis-counters, department stores must reconsider the best way to secure and maintain a com-petitive advantage. Mervyn’s, a leading cloth-ing retailer with more than 280 stores in the United States, uses a sophisticated decision support system (DSS) to help its analysts to identify trend-right products and facilitate quick decisions.

Although existing systems at Mervyn’s provided large amounts of data, the challenge was to construct a system that would effectively integrate and distill that data into mission-critical information for decision making. Inade-quate data access and analysis capabilities in their earlier systems prevented managers from exploiting the wealth of information buried in Mervyn’s transaction data.

Mervyn’s need for quick access to 300 to 700 gigabytes of data compounded this problem. Performing sophisticated analyses within a rea-sonable time frame on a data set of this size was beyond the scope of most existing decision sup-port technologies. The solution required query performance optimization and various parallel software schemes new to decision support.

Mervyn’s also needed to standardize business analyses for more meaningful com-parisons. Sue Little, manager of Merchandising Planning and Logistics Systems at Mervyn’s, says, “We’ve never had data where units and dollars matched. Buyers looked at everything from a dollar perspective and Inventory Man-agement saw everything from a unit perspec-tive. There was never anything that tied the two kinds of numbers together easily.”

To solve these problems, MicroStrategy, a manufacturer of DSS development software, and Mervyn’s used DSS Agent (a development application manufactured and sold by Micro-Strategy) to develop the Decision Maker’s Workbench (DMW). This DSS application allows for trend, performance, and inventory stock analyses. Analysts can see, by ad zone, how product sales peak and valley over seasons, or how they vary from region to region. These data help them decide when and where to peak or deplete inventory. Through its use of sym-metric multiprocessing hardware and parallel query processing technology, DMW reduced the time required to perform this task from an hour to less than a minute, helping Mervyn’s realize its strategic aim of securing a competitive advantage through timely decision making.

Sue Little comments, “DMW let us look at different types of merchandise together, online, at the atomic level, in either dollars or units. We’re finally comparing apples with apples, and we’re now spending only 10 percent of our time gathering data and 90 percent acting upon it, rather than the other way around.”

The DMW application incorporates fea-tures such as data surfing, drill down, intelligent agents, and alerts. Mervyn’s application includes maps that display geography-based exception reporting and executive views that summarize business trends and exceptions in a newspaper-like interface. DMW includes forecast-ing capabilities, enabling end users to develop what-if scenarios by performing ad hoc analyses on query results.

The study of decision support systems (DSSs for short) is not about computers. Although they play an integral role in the DSS world, computers are just one part of the pic-ture. The study of DSSs is really about people—about how people think and make decisions, as well as how they act on and react to those deci-sions. The DSS field is one of the few informa-tion systems (IS) fields that can claim both identifiable roots and a clear focus. Decision support systems are designed, built, and used to assist in the activity that they are named for: supporting the decision-making process. They are not, however, intended to make the deci-sions themselves, although we will see examples of DSSs that do almost exactly that. The real purpose of a DSS is to provide support to the decision maker during the process of making a decision. Therefore, when you study DSSs, you study people, decisions, and how those decisions are made.
1-1: DSS DEFINED

COMMON CHARACTERISTICS

This section offers a definition of decision support system, but first we need to look at some characteristics common to most, if not all, DSS applications. Table 1-1 contains a list of those attributes.

As you can see, the typical DSS serves many functions, which makes arriving at a simple but comprehensive definition of a DSS more difficult than it may first appear. In fact, you can probably find as many definitions as you can find books on the subject. The good news is that the various definitions address some common themes, and a focus on those themes should yield a good working definition of the concept.

The first of those themes is problem structure. This dimension focuses on the degree to which a decision or decision-making situation displays certain structural characteristics. For example, in the traditional view of this dimension, a highly structured decision situation includes easily determined objectives that are not subject to conflict, clearly defined or select alternative courses of action, and ascertainable outcomes. Conversely, in a highly unstructured decision the objectives of the situation often conflict, the alternatives available to the decision maker are difficult to isolate, and the effect of a particular course of action or selection of an alternative carries with it a high degree of uncertainty. The role of the DSS is to provide support to the decision maker on the “structurable” portions of the decision. With this support, the decision maker is free to focus his or her cognitive resources on the truly unstructured portions of the problem—those portions that, given the limits of technology to execute the complex problem-solving strategies contained in human memory, are better left for resolution by human decision makers. The decision processes employed in addressing the unstructured portions of a decision situation can be thought of as the human processes we do not yet understand well enough to effectively simulate via automation. More on that will be discussed later.

A second theme found in most definitions of a DSS centers around the decision outcome. The information-rich environment of today’s organization is where the DSS

<table>
<thead>
<tr>
<th>TABLE 1-1 Common DSS Characteristics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Employed in semistructured or unstructured decision contexts</td>
</tr>
<tr>
<td>Intended to support decision makers rather than replace them</td>
</tr>
<tr>
<td>Supports all phases of the decision-making process</td>
</tr>
<tr>
<td>Focuses on the effectiveness of the decision-making process rather than its efficiency</td>
</tr>
<tr>
<td>Is under control of the DSS user</td>
</tr>
<tr>
<td>Uses underlying data and models</td>
</tr>
<tr>
<td>Facilitates learning on the part of the decision maker</td>
</tr>
<tr>
<td>Is interactive and user-friendly</td>
</tr>
<tr>
<td>Is generally developed using an evolutionary, iterative process</td>
</tr>
<tr>
<td>Provides support for all levels of management from top executives to line managers</td>
</tr>
<tr>
<td>Can provide support for multiple independent or interdependent decisions</td>
</tr>
<tr>
<td>Provides support for individual, group, and team-based decision-making contexts</td>
</tr>
</tbody>
</table>
is commonly found and where it performs much of its work. A key element of a technology that is intended to support the process associated with making a decision is the decision itself. We will spend a great deal of our time considering exactly what decisions are and why we need technology to support us in making them. The effectiveness of a given decision, or the degree to which the decision succeeds in reaching its objectives, is an essential element in the decision-making process. Therefore, our definition of a DSS must take into account the role the system plays in supporting decision effectiveness.

A third theme present in definitions of a DSS concerns managerial control. The ultimate responsibility for the outcomes associated with decisions lies with the manager. When you stop to think about it, the decision can be argued to be the manager’s most powerful tool. Through decision implementation, all the organization’s resources are deployed or structured at any given moment. The decision acts as the primary mechanism to reach the organization’s strategic objectives and to fuel its successes. Whether the decision is thought of as a choice, a course of action, or a strategy, decision activity results in the selection of and commitment to one of multiple alternatives. The control of this final selection lies with the decision maker(s). To that end, the DSS must provide support to the selection process but, ultimately, must be positioned to allow the final selection to be made by those managers directly responsible and accountable for the outcomes.

Based on these three common themes, we can devise a formal definition of a DSS that will serve as a foundation upon which a more detailed understanding of the construction, application, and use of DSSs can be built. A decision support system is a system under the control of one or more decision makers that assists in the activity of decision making by providing an organized set of tools intended to impose structure on portions of the decision-making situation and to improve the ultimate effectiveness of the decision outcome.

WHAT IT CAN AND CANNOT DO

The DSS clearly offers management a powerful tool and is rapidly becoming an integral component of managerial work. The speed with which today’s information becomes yesterday’s news continues to increase at a staggering rate. Tomorrow’s manager will confront an ever-narrowing window of opportunity within which effective decisions will need to be made. Deadlines will be measured in days, hours, and minutes rather than in quarters, months, and years. The leveraging of technology that will allow tomorrow’s manager to be effective in such a high-speed environment is what decision support is all about. To meet the demands of managerial work, a DSS must be able to provide the decision maker with certain key elements vital to his or her success. A DSS cannot offer benefits in all decision situations or by all decision makers, however. Its effectiveness depends on the degree of fit between the decision maker, the context of the decision, and the DSS itself. Assuming the fit is there, however, we can, in general, expect several potential benefits from using a DSS. In addition, we must acknowledge and understand the limitations of using a DSS. Table 1-2 contains a list of such potential benefits and limitations.

As you can see, the DSS is expected to extend the decision maker’s capacity in processing the mountain of information involved in making a decision. Further, many components of a decision situation, although structured, are nonetheless highly com-
TABLE 1-2  Benefits and Limitations of DSS Use

Benefits
• Extend the decision maker’s ability to process information and knowledge
• Extend the decision maker’s ability to tackle large-scale, time-consuming, complex problems
• Shorten the time associated with making a decision
• Improve the reliability of a decision process or outcome
• Encourage exploration and discovery on the part of the decision maker
• Reveal new approaches to thinking about a problem space or decision context
• Generate new evidence in support of a decision or confirmation of existing assumptions
• Create a strategic or competitive advantage over competing organizations

Limitations
• DSSs cannot yet be designed to contain distinctly human decision-making talents such as creativity, imagination, or intuition
• The power of a DSS is limited by the computer system upon which it is running, its design, and the knowledge it possesses at the time of its use
• Language and command interfaces are not yet sophisticated enough to allow for natural language processing of user directives and inquiries
• DSSs are normally designed to be narrow in scope of application, thus inhibiting their generalizability to multiple decision-making contexts

plex and time-consuming. The DSS can solve those portions of the problem, and save on cognitive resources and, more importantly, large blocks of precious time for the decision maker. As a result, using a DSS can be expected to decrease the overall time involved in reaching a complex, unstructured decision.

Additional benefits can be found in the areas of innovation and creativity. Simply using the DSS can provide the decision maker with potential alternatives that might otherwise go unnoticed or appear too complex and difficult to pursue. The tools within the DSS can stimulate the problem solver to reach innovative insights regarding solutions and their associated outcomes. In addition, the output of the DSS may often justify the position of the decision maker(s), thus facilitating consensus among stakeholders. Finally, given the shrinking window of opportunity associated with the pace of business, the DSS may provide competitive advantage to organizations. To achieve some or all of these potential benefits, however, the manager must understand not only the appropriate application of a particular decision support tool but also its limits.

No matter how well a DSS is designed, its value is constrained by certain limitations. To begin with, the DSS, like any other computer-based system, contains only the “knowledge” given to it by its designers, and it possesses only the specific “skills” associated with its tool set. Although we will explore the concept of a DSS that can “think” in Chapter 7, and of one that can “learn” in Chapter 9, the DSS is still limited by its design and its designers.

Other drawbacks of the DSS include limits on its ability to perform reasoning processes that require distinctly human characteristics such as creativity, intuition, or imagination. Such cognitive activities still belong to human experience and do not lend themselves well to automation or machine simulation. Further, the DSS must be
designed to communicate its information to its user in an understandable manner that is useful within the context of the decision situation. Although humans can adapt their methods of communication readily to a given situation, computer systems such as DSSs cannot. Therefore, the methods by which we communicate with a DSS and those by which it responds may limit its effective use. We will explore these issues of interface and interaction in Chapter 15.

Finally, and maybe most important, is the understanding that a “universal DSS” does not exist and probably never will. A typical DSS is designed to be useful within a relatively narrow scope of problem-solving scenarios. Thus, to effectively solve a complex problem or reach a decision of significant magnitude may require the use of several DSSs. In that case, the decision maker must coordinate multiple systems that may require output from one as input to another. The integration of multiple DSSs then becomes, in and of itself, a complex and uncertain decision scenario.

In summary, decision support systems can make the decision process more effective for the human decision maker. They cannot, however, overcome or prevent the actions of a poor decision maker. The user ultimately controls the process and must understand when to use a DSS, what DSS(s) to use, and, most importantly, to what degree to depend on the output and information obtained from the DSS. The manager must see the DSS as a valuable tool in the decision process rather than as a mechanism that makes the decision.

1-2: HISTORY OF DECISION SUPPORT SYSTEMS

THE EVOLUTION OF THE DSS

The roots of the DSS grew out of the application of quantitative models to the daily problems and decisions managers faced in an organizational environment. The concept was born in the early 1970s and is generally attributed to two articles written during that time. The first, written by J. D. Little (1970), was entitled “Models and Managers: The Concept of a Decision Calculus.” Little observed that the biggest problem with management science models was that managers rarely used them. He described the concept of a decision calculus as a “model-based set of procedures for processing data and judgments to assist a manager in his decision making” (p. B470). Little suggested that for such a system to succeed, it must be simple, robust, easy to control, complete on issues of importance, adaptive to the needs of its user, and easy to communicate with.

The second article, “A Framework for Management Information Systems,” was written by Gorry and Scott Morton (1989). In that article, they coined the term decision support system and developed a two-dimensional framework for computer support of managerial activities (see Table 1-3). Each of these dimensions within the framework is assumed to be continuous rather than composed of discrete components. The vertical dimension represents a classification of decision structure as originally proposed by Simon (1960). Simon suggested that decisions be categorized according to the degree to which they are programmed (repetitive, routine, and commonplace) or nonprogrammed (novel, unique, and consequential). The horizontal dimension of the framework represents the levels of managerial activity proposed by Anthony (1965).
TABLE 1-3 Gorry and Scott Morton’s Framework for Decision Support

<table>
<thead>
<tr>
<th>Management Activity</th>
<th>Type of Decision</th>
<th>Operational Control</th>
<th>Management Control</th>
<th>Strategic Planning</th>
<th>Support Needed</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Structured</td>
<td></td>
<td></td>
<td>Load balancing for production lines</td>
<td>Physical plant location</td>
<td>MIS, quantitative models</td>
</tr>
<tr>
<td>Semistructured</td>
<td></td>
<td>Securities trading</td>
<td>Establishing marketing budgets for new products</td>
<td>Analysis of acquisition of capital assets</td>
<td>DSS</td>
</tr>
<tr>
<td>Unstructured</td>
<td>Determining the cover photo for a monthly magazine</td>
<td>Hiring of new managerial personnel</td>
<td>Determination of research and development projects</td>
<td>Human reasoning and intuition</td>
<td></td>
</tr>
</tbody>
</table>

Here, all managerial activities can be categorized into three unique classes: (1) strategic planning, where decisions related to the objectives of the organization are made; (2) management control, where decisions related to the effective and efficient procurement and deployment of the organization’s assets are made; and (3) operational control, where decisions relating to the day-to-day tasks and activities of the organization are made. Combining the approaches of Simon and Anthony produces a framework to guide the allocation of IS resources to where the greatest payoff, or return on investment, may be found. From this framework, the concept of the decision support system was born.

The evolution of the DSS from its conception in the 1970s to the present day includes numerous extensions of the original notion. The modern-day study of DSSs must include a focus on conventional model-based systems, knowledge-based systems, artificial intelligence, expert systems, executive information systems, group support systems, data visualization systems, and organizational decision support systems. Each of these areas will be explored in depth in later chapters.

A BRIEF GLIMPSE OF THE FUTURE

If the number and types of DSSs and associated applications are any indication, then the DSS may well be the most successful of all computer-based information systems. Today’s organizations recognize the need for embracing information technology as an integral tool in implementing their strategic plans. Tomorrow’s organization will employ computer-based IS in virtually every aspect of its operation—from the development and marketing of its product or service to the coordination of its employees, processes, and activities worldwide. The role of the DSS in tomorrow’s organization cannot be overstated. Managers will rely on the availability of more powerful and useful DSS applications in conducting their daily activities. The DSS of the twenty-first century will serve as the primary vehicle for keeping up with the seemingly exponential growth in the size, complexity, and speed of business activities. In Chapters 17 and 18, we will explore the various applications and uses for DSS technologies in the global marketplace of the future.
CHAPTER 1 ♦ Introduction to Decision Support Systems

1-3: INGREDIENTS OF A DSS

BASIC DSS COMPONENTS

It should be clear to you by now that a DSS is not a simple system with common, identifiable characteristics and a singular or common purpose. As we have seen, simply defining a DSS requires consideration of numerous factors including its intended purpose, the context within which it will be used, and its outcome objectives. Describing or classifying a DSS in terms of its components poses an equally challenging task. Once again, before we can construct a generalized component structure for a DSS we need to look at the various methods of classifying DSSs by their component parts.

An early classification by Alter (1980) divided the DSS components into seven categories, depending upon the degree of direct influence each component part could have on a given decision. Figure 1-1 shows the classification scheme proposed by Alter along with a mapping of the various components to common problem-solving and decision-making tasks. Originally, Alter’s classification assumed that each category could be thought of as a separate and distinct system rather than as a functional component within a larger system. As Figure 1-1 shows, however, the seven separate components can be logically grouped into a simpler classification scheme of either model-oriented or data-oriented systems. We will soon see that this simple dichotomy reflects the more modern classification methods for DSSs.

Another approach to isolating and defining individual components of a DSS focuses on the nature of the language provided by the DSS to manipulate data or models and the degree to which the language used is procedural or nonprocedural. The concept of language procedurality is relatively simple and is unique to the design and application of DSSs. Basically, the user of a DSS should be able to specify the information he or she wants from that DSS, as well as whether that information exists in a database, must be derived or computed by one or more models contained within the DSS, or must be computed by a model constructed by the DSS from other models stored specifically for that purpose. Languages that are highly procedural require the user to be specific with regard to how the data are to be obtained, from where they are to be

![Figure 1-1: Alter’s Classification of DSS Components](image-url)
obtained, and exactly how a particular model or set of models is to process that data. Nonprocedural languages simply require the user to specify the necessary information, and the DSS takes care of the rest. Somewhere between these two extremes we find a degree of procedurality that allows users a means of providing parameters to prespecified data retrieval operations or a predetermined set of models. As you can see, the degree of procedurality of the language system allows for a wide variety of configurations and designs.

Building upon the two classification methods discussed, the components of a DSS can generally be classified into five distinct parts:

1. The data management system
2. The model management system
3. The knowledge engine
4. The user interface
5. The user(s)

THE DATA MANAGEMENT SYSTEM

The data management component of a DSS retrieves, stores, and organizes the relevant data for the particular decision context. Additionally, the data management system provides for the various security functions, data integrity procedures, and general data administration duties associated with using the DSS. These tasks are carried out within the data management component by several subsystems. Among these subsystems are the database(s), database management system, data repository, and data query facility. Each of these elements is discussed in greater detail in Section 1–4.

THE MODEL MANAGEMENT SYSTEM

Similar to the role of the data management system, the model management component performs the retrieval, storage, and organizational activities associated with the various quantitative models that provide the analytical capabilities for the DSS. Within this component are the model base, model base management system, model repository, model execution processor, and model synthesis processor. A detailed review and comparison of this component to the data management system is performed in Section 1–4.

THE KNOWLEDGE ENGINE

The knowledge engine performs activities related to problem recognition and generation of interim or final solutions, as well as other functions related to the management of the problem-solving process. The knowledge engine supplies the “brains” of the system. The data and the models come together here to provide the user with a useful application that supports the decision context. We will look at the knowledge engine in greater detail in Section 1–5.

THE USER INTERFACE

As in any other computer-based information system, the design and implementation of the user interface is a key element in DSS functionality. The data, model, and processing components of the DSS must be easily accessed and manipulated if the DSS is to
provide the necessary support to the decision context without getting in the way of the task at hand. Furthermore, the ease with which the user can communicate with the DSS, whether for specifying parameters or investigating the problem space, is crucial to the effectiveness of a DSS. These responsibilities, among many others, fall to the user interface and will be discussed in Section 1–6.

THE DSS USER

The design, implementation, and use of a DSS cannot be effective without considering the role of the user. User skill set, motivations, knowledge domain, patterns of use, and role(s) within the organization constitute the essential elements in the successful application of a DSS to a decision context. Remember, one of the primary characteristics of a DSS is user control. Without considering the user as a part of the system, we are left with a set of computer-based components that, by themselves, provide no useful function at all. We look at the user more closely in Section 1–7 and again in Chapter 2.

1-4: DATA AND MODEL MANAGEMENT

THE DATABASE

As more and more organizations realize the importance of data as a corporate asset that must be managed, the processes by which data are collected, stored, and disseminated continue to improve. This increased focus on the value of data to the organization is particularly relevant to the study of decision support systems because the quality and structure of the DSS database component largely determine the success of the modern DSS.

A database is an integrated collection of data, organized and stored in a manner that facilitates its easy retrieval. The structure of a database should correspond to the needs of the organization and should allow for access by multiple users and, when appropriate, for use by more than one application. A database organizes data into a logical hierarchy based on the degree of aggregation or granularity of that data. This hierarchy consists of four elements:

1. Database
2. Files
3. Records
4. Data elements

Figure 1-2 shows how the various elements of the hierarchy are logically integrated in a typical database.

Just as the database is a collection of integrated data organized into files, the file serves as a collection of data organized into records that all relate to a particular information focus. For example, a particular data file may contain information regarding sales transactions during a specific period, say for an organization's last fiscal year—or a similar data file may contain historical sales volume information for an entire industry over the last 5 years. Regardless of the nature of the data, it must all be related to a common subject and must be organized so that various useful aggregations of the data within that file can be created and compared.
Although data within files are organized into a common structure, the sources of the data contained within the files may be quite diverse. The files may include data generated from internal transactions, external sources, and individual users of the DSS.

Internal data normally come from an organization’s daily transaction processing systems. Payroll and salary data, sales transactions, production level and throughput, as well as other transactions generated by the various business units of the organization, are all examples of internal data sources that may appropriately contribute one or more files to a typical DSS database. Transaction-level data are a major source of information regarding the operations of an organization or business unit. Depending on the problem context in which the DSS is operating, these data may be all that is necessary for the DSS to provide the necessary level of decision support. Oftentimes, however, an internal data source is only one contributor of information to the DSS database.

External data sources are as broad as the imagination allows. They may include targeted collections of data for a particular industry, market research data, employment information for a particular geographic region, rate schedules or cost tables for particular products or product classes, geopolitical economic or census data, or a historical list of all court cases within a particular legal jurisdiction, such as the Supreme Court. The list is literally endless. It is a safe bet that if a source of data generation exists, then a database containing that data also exists. The good news for DSS designers and users is that most of the imaginable external data can be obtained with relative ease and at little or no cost. The bad news is that, with all of the data out there and with more being made available each day, just figuring out where to look can be a monumental task. Fortunately, however, developments such as the Internet and World Wide Web make access to external data much easier than in the past. In addition, data delivery services such as Mead Data Central’s LEXIS-NEXIS, Dow Jones News Retrieval, CompuServe, and a myriad of others aid the downloading, integration, and analysis of external data for use in the corporate DSS environment.
Regardless of their source, the data contained within the files in the DSS database can be organized into homogenous structures, or subunits, called records. In turn, the data contained within each record are organized into a series of data elements, or fields. The data element is the smallest unit of decomposition within the logical hierarchy and cannot normally be broken down further. Each record is constructed so that it contains a set of data elements related to a specific instance of the type of information contained within the file. An example might begin with a file containing data about the sales of various items in an organization’s product line. Information such as product code, product name, date and time of sale, quantity or amount of sale, location of sale, and other specific pieces of data concerning the transaction may be organized and stored within a data record. Once these records are stored, they can be retrieved individually based on a unique piece of data, such as invoice number, or they can be retrieved as a unique subset, such as all records containing a particular product code. More importantly, they can be combined with other sources of organized data within the DSS database to allow for extensive analysis of relationships that might otherwise go unnoticed by the DSS user. Through its ability to combine data records from multiple sources, the DSS increases the value of the data within the decision-making process.

One additional source of data that may be contained within the typical DSS database is referred to as individual-level or private data. Each instance of use of a DSS generates data about the decisions made, the questions asked, and the various combinations of data constructed for that particular problem context. In addition, individual heuristics, or rules of thumb, used by various DSS users may also be stored in the DSS database. These individual-level data elements and records often create a unique personality for the DSS, thus tailoring the system to the needs of a specific user or group of users within its defined problem context. Information such as an individual’s personal assessment of specific data or past situations, a heuristic stating that “during December, sales of Product A can be expected to be 37 percent of the sales of Product B,” or simply a library of past queries or data source integrations may all be stored as individual-level data in the DSS database.

THE DATABASE MANAGEMENT SYSTEM

The multitude of data that can be organized into files and databases must be managed, and this important role falls to the database management system (DBMS). The DBMS has two main responsibilities:

1. The coordination of all tasks related to storing and accessing information in the database and disseminating information to the community of DSS users
2. The maintenance of logical independence between the data contained in the DSS database and the DSS application

Modern DBMSs possess a wide variety of capabilities and are generally managed by a skilled database administrator assigned specifically to that task. Commercial DBMS packages from Sybase, Oracle, IBM, and others provide powerful foundational applications from which the DSS database can be managed. Recent DBMS developments include facilitating the integration of a large number of unrelated, or disparate, data sources into a single, accessible database known as a data warehouse.
houses provide large amounts of data to the DSS in a form and manner that is more conducive to DSS use. They will be discussed in greater detail in Chapters 10, 11, and 12.

The first responsibility of the DBMS—the coordination of tasks related to the storage, access, and dissemination of database information— involves several activities and functions. These functions include updating (in the form of adds, deletes, edits, or changes) data records and elements as transactions occur, facilitating the integration of data from various sources, and retrieving data from the database for queries or report generation. In addition, the DBMS performs many administrative functions related to DSS operation such as data security (control of unauthorized access, database error recovery, data integrity, and concurrency issues), the creation of personal or temporary databases for individual user experimentation and analysis, and the tracking of DSS usage and data acquisition, among others. Many of these activities are performed in the background and are intended to be invisible to the DSS user. Nonetheless, they are essential to the successful use of a DSS in a problem-solving context.

The second main responsibility of the DBMS, that of maintaining logical independence between the data and the DSS application, is also crucial to the success of a DSS. The typical DSS scenario involves one or more users making decisions that are largely based upon information taken from a large pool of constantly evolving data. The DBMS must manage the physical organization and structuring of this data within the database and its associated files without constraining the logical arrangement of the data to either the DSS application or the DSS user(s). The same data may need to be presented in one form, such as a graph or table, under one set of circumstances and in a completely different manner, such as input to an analytical model, in another situation. In addition, as new sources of data are made available to the DSS, the DBMS must manage the integration of these disparate sources so that they appear to be all neatly organized in a common structure and location even when they are not. By maintaining independence between physical data structure and DSS applications, the DBMS allows for a much broader use of a single database, either by multiple DSS applications, multiple user groups, multiple problem contexts, or any combination thereof. Table 1-4 describes the various functions of a typical DBMS.

THE MODEL BASE

A model is a simplification of some event or process constructed for the purpose of studying that event and thus developing a better understanding of it. As a simplified form of reality, a model is intended to resemble the actual process or event as closely as possible but normally does not contain the same level or kind of detail as the phenomenon itself. In many cases, we can ascertain from the model many characteristics of a process or event. Moreover, we can often predict the nature or outcome of that event under certain conditions without having to actually experience or re-create the event under study. The value of this capability can be found in the reduced cost, effort, and time derived from studying a model rather than the event itself. Imagine the effort (and risk) necessary to remain inside a hurricane for its duration in order to predict its path or its strengthening. Imagine the costs associated with having to build an airplane to see how fast it will fly, or if it will fly at all. Imagine the time it would take to determine how popular a new product or service is going to be if you had to ask everybody's opinion about it before you began selling it. In each of these cases, modeling can be an
### TABLE 1-4 General Functions of the DBMS

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Data Definition</strong></td>
<td>Provides a data definition language (DDL) that allows users to describe the data entities and their associated attributes and relationships</td>
</tr>
<tr>
<td></td>
<td>Allows for the interrelation of data from multiple sources</td>
</tr>
<tr>
<td><strong>Data Manipulation</strong></td>
<td>Provides the user with a query language to interact with the database</td>
</tr>
<tr>
<td></td>
<td>Allows for capture and extraction of data</td>
</tr>
<tr>
<td></td>
<td>Provides rapid retrieval of data for ad hoc queries and reports</td>
</tr>
<tr>
<td></td>
<td>Allows for the construction of complex queries for retrieval and data manipulation</td>
</tr>
<tr>
<td><strong>Data Integrity</strong></td>
<td>Allows the user to describe rules (integrity constraints) to maintain the integrity of the database</td>
</tr>
<tr>
<td></td>
<td>Assists in the control of erroneous data entry based on the defined integrity constraints</td>
</tr>
<tr>
<td><strong>Access Control</strong></td>
<td>Allows identification of authorized users</td>
</tr>
<tr>
<td></td>
<td>Controls access to various data elements and data manipulation activities within the database</td>
</tr>
<tr>
<td></td>
<td>Tracks usage and access to data by authorized users</td>
</tr>
<tr>
<td><strong>Concurrency Control</strong></td>
<td>Provides procedures for controlling simultaneous access to the same data by more than one user</td>
</tr>
<tr>
<td><strong>Transaction Recovery</strong></td>
<td>Provides a mechanism for restart and reconciliation of the database in the event of hardware failure</td>
</tr>
<tr>
<td></td>
<td>Records information on all transactions at certain points to enable satisfactory database restart</td>
</tr>
</tbody>
</table>

effective source of information for the decision process. Although the underlying models contained within a DSS are the essence of the concept of decision support, the degree to which these models are designed and constructed to resemble the real thing normally depends on the needs of the decision maker using them. Nonetheless, the lack of one or more models associated with the problem context under investigation limits a DSS’s contribution. We will focus on the various types of models and their construction in detail in Chapter 4.

The model base in a DSS is the modeling counterpart to the database. Just as the DSS database stores the data used by the DSS, the model base contains the various statistical, financial, mathematical, and other quantitative models the DSS uses to perform a variety of analyses. Its model base differentiates a DSS from other computer-based information systems. The ability to run individual or combined models or to construct new models makes the DSS a powerful support tool in the problem-solving environment.
MANAGING THE MODELS

The underlying models in a typical DSS can range in number, size, and complexity much like the data stored in a DSS database. To manage numerous analytical tools, the DSS uses a model base management system (MBMS). Table 1-5 contains a list of the basic functions of the MBMS.

Two important responsibilities of the MBMS are the execution and integration of the models available to the DSS and the modeling of user preferences. The user is normally required to provide data as input to a model or to provide certain parameters that may affect the model’s execution. The MBMS takes care of this process by providing the user with easy access to the various models as well as facilitating entry of important parameters and data to the models.

Often to execute one or more models, the user must provide specialized syntax or commands to initiate the process. The MBMS generates the necessary command structure and locates the models for the DSS. The specific formatting requirements for data or parameter entry are also handled by the MBMS subsystem. Finally, the need to combine or integrate several individual models into a more complex one occurs regularly in DSS use. The MBMS facilitates this integration by providing the necessary links between the models as well as controlling the order of model sequencing and execution.

The modeling of user preferences is the process of collecting, organizing, and integrating the preferences, judgments, and intuitions of the individual DSS user into the modeling operation. Decision makers and decision contexts both exhibit constantly changing and conflicting preferences. The MBMS, where appropriate, provides a mechanism to incorporate the individual user or problem context preferences into the analytical process so that the outcomes represent not only the quantitative, or stochastic, results but also the more qualitative, or context-specific, conditions and constraints. In short, the MBMS plays an important role in facilitating the decision support process through a DSS application. We explore this issue of user characteristics and preferences in Chapter 2.

TABLE 1-5  General Functions of the MBMS

<table>
<thead>
<tr>
<th>Modeling Language</th>
</tr>
</thead>
<tbody>
<tr>
<td>Allows for the creation of decision models from scratch or from existing modules</td>
</tr>
<tr>
<td>Provides a mechanism for the linking or chaining of multiple models to allow for sequential processing and data exchange</td>
</tr>
<tr>
<td>Allows the user to modify models to reflect specific preferences</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Model Library</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stores and manages all models and solver algorithms for easy access and manipulation</td>
</tr>
<tr>
<td>Provides a catalog and organizational schema of stored models along with brief descriptions of their individual functions or applications</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Model Manipulation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Allows for management and maintenance of the model base with functions similar to those found in a DBMS (i.e., run, store, query, delete, link, etc.)</td>
</tr>
</tbody>
</table>
All decision or problem-solving contexts require reasoning. A decision that can be made without reasoning is really not a decision at all. Admittedly, the more structured a decision context is, the less reasoning is necessary for a successful outcome. It follows that we can envision a point at which the reduction in required reasoning converges with the increase in problem structure. At this intersection we find those decisions that are so structured that no reasoning is required to make them. By default then, it must be the point where no decisions exist.

At this stage, it is critical that we clearly express what reasoning is. **Reasoning** is the process by which new information is derived from a combination or combinations of existing, or previously derived, information. In its simplest form, reasoning allows us to rely on information as facts, even though we haven’t specifically verified that information personally. For example, if we know for a fact that a particular company has a current ratio (current assets to current liabilities) of 2.6 and a quick ratio (liquid assets to current liabilities) of 1.7, we know two isolated facts about this company. Through the process of reasoning, however, we also can “know” that the liquidity of the company is generally sound. No one told us this piece of information, nor did we personally verify it, but we know it nonetheless. Many forms of reasoning are used in problem solving and decision making, and we will explore them in greater detail in Chapters 7, 8, and 9. At this point, however, our definition is sufficient to aid in our understanding of yet another component of a DSS: the knowledge base.

The knowledge base is where the “knowledge” of the DSS is stored. By **knowledge**, we mean the rules, heuristics, boundaries, constraints, previous outcomes, and any other information that may have been programmed into the DSS by its designers or acquired by the DSS through repeated use.

The information contained within the knowledge base component of a DSS bears unique characteristics that differentiate it from information contained in either the database or the model base components. First, the knowledge base contains information that is generally problem domain-specific; that is to say, it is knowledge that is applicable or relevant within only a narrow problem-solving context. Conversely, the database and model base components store a wide range of domain-related elements. The database or databases associated with a DSS are not limited to a particular problem-solving domain. The models contained within the model base are likewise not constrained by the defined problem-solving context within which the DSS resides. The knowledge base, however, generally does not contain anything that is not directly relevant to, or directly derived from, the problem-solving context. Think of it this way: Knowledge is a domain-specific construct whereas data and models are useful across several domains or tasks.

So what exactly is contained in the DSS knowledge base? All kinds of knowledge that might be used by a domain expert reside within a knowledge base: descriptions of various objects or entities and their relationships; descriptions of various problem-solving strategies or behaviors; domain-related constraints, uncertainties, and probabilities; and so on. The knowledge contained in the knowledge base can be categorized into two simple groups: facts and hypotheses. The **facts** represent what we know to be true at a given time. The **hypotheses** represent the rules or relationships that we believe exist between the facts.
Let’s assume that we have a simple knowledge base that contains a select number of variables deemed by experts to be most important in evaluating a potential borrower’s creditworthiness:

1. Number of years of credit history (a numerical fact)
2. Quality of the credit history (a ranking fact: perfect, almost perfect, very good, good, not good, awful)
3. Job description of the borrower (a category fact: doctor, lawyer, computer operator, college professor)
4. Number of years of employment in this job area (a numerical fact)
5. Total income of the borrower (a numerical fact)
6. Total debt obligations of the borrower (a numerical fact)
7. Amount of money the borrower wants to borrow (a numerical fact)

Now, in addition to these facts, our knowledge base also contains the relationships among them. In other words, we have also stored the manner in which one or more facts are related to each other. Figure 1-3 graphically illustrates the relationships between the facts contained within our knowledge base.

In this case, our objective is to make a decision regarding whether we should lend money to a specific person. Notice, however, that the relationships between the facts

---

**FIGURE 1-3** Examples of Credit Rule Combinations

<table>
<thead>
<tr>
<th>Credit Rule #1:</th>
</tr>
</thead>
<tbody>
<tr>
<td>If credit history is &lt; 1</td>
</tr>
<tr>
<td>AND</td>
</tr>
<tr>
<td>quality of credit history is not good</td>
</tr>
<tr>
<td>OR</td>
</tr>
<tr>
<td>quality of credit history is awful</td>
</tr>
<tr>
<td>OR</td>
</tr>
<tr>
<td>employment years is &lt; 1</td>
</tr>
<tr>
<td>OR</td>
</tr>
<tr>
<td>total income &lt; total debt</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Credit Rule #2:</th>
</tr>
</thead>
<tbody>
<tr>
<td>If credit history is &gt; 1</td>
</tr>
<tr>
<td>AND NOT</td>
</tr>
<tr>
<td>quality of credit history is not good</td>
</tr>
<tr>
<td>OR NOT</td>
</tr>
<tr>
<td>quality of credit history is awful</td>
</tr>
<tr>
<td>AND</td>
</tr>
<tr>
<td>total income &gt; total debt</td>
</tr>
<tr>
<td>AND</td>
</tr>
<tr>
<td>employment years is &gt; 3</td>
</tr>
<tr>
<td>AND</td>
</tr>
<tr>
<td>total income is &gt; $50,000</td>
</tr>
<tr>
<td>AND</td>
</tr>
<tr>
<td>amount to borrow is &lt; $25,000</td>
</tr>
</tbody>
</table>

---

Credit Risk Is High

Credit Risk Is Low

Make Loan

---
are not focused on any particular person but rather are simply stated in the form of rules. One rule may state the relationship between fact 1 and fact 2, another rule may state the relationship between fact 5 and fact 6, and so on. Although these rules can be thought of as a series of models, they are unique in that they can be combined in any manner and in any order to produce a decision outcome. Figure 1-3 shows two such combinations. For example, let’s say the borrower has a short credit history (high risk), but the history that exists is perfect (extremely low risk). She has a good job and has been employed in this job area for a long time (both low risk). However, her debt is extremely high compared to her income (high risk) and the amount of money she wants to borrow is also extremely large (high risk). In this case, the knowledge base combines the facts we have presented with the relationships among those facts (relationships, by the way, that we programmed into the DSS) and might help us conclude that we should decide against making the loan to the borrower. Bear in mind that this example is a simple one and, when compared to the types and sizes of typical knowledge bases, a somewhat trivial one as well. It demonstrates, however, just how the knowledge base contributes to the DSS as a whole.

KNOWLEDGE ACQUISITION

By now, you should be wondering where all this knowledge and all these rules come from. If the user of the DSS already knows this stuff then why do we need a DSS? Well, that’s the point! The typical user of a DSS does not know this stuff. The DSS user relies on the integrity of the data in the database, the quality and accuracy of the models in the model base, and the merit and reliability of the facts and relationships in the knowledge base when using the DSS during a decision-making activity. So how does all that knowledge get into the DSS?

One or more people called knowledge engineers (KEs) interview the domain experts and gather the information necessary for the knowledge base. Knowledge engineers are specially trained to interact with domain experts for the purposes of acquiring all of the expert’s knowledge in a particular domain and all of the relationships among that knowledge. A variety of knowledge acquisition techniques such as interviewing, protocol analysis (thinking out loud), and modeling, among many others, are used to get the information out of the experts’ minds and into the knowledge base. The process of knowledge acquisition is a daunting task, indeed. We will focus on the different techniques associated with acquiring and organizing the knowledge of experts in Chapters 7 and 8.

KNOWLEDGE RETRIEVAL

Once the facts and relationships are collected and inserted into the knowledge base, we need a method of getting them back out in an organized and useful fashion. The inference engine (IE) is that part of the knowledge base component that facilitates this process. The IE is a program module that activates all the gathered domain knowledge and performs inferencing, or basic reasoning, to work toward a solution or conclusion based upon the values for the facts given and the relationships or rules associated with them. The inference engine must be supplied with rules on how to apply the rules, with strategies for conflict resolution in the event that two rules oppose each other, and in
many cases, with methods of determining the probability that a conclusion derived from a set of facts and rules is reliable. We will discuss IEs in depth in Chapter 7.

It should be clear that the knowledge base component of a DSS, when combined with the capabilities of modern DBMS and MBMS facilities, creates a potent tool for supporting the complex decisions managers must make. These three components by themselves, however, can accomplish little. They need to be accessible to the DSS user in a manner that truly supports the decision-making process and facilitates their individual and combined use with a minimum of effort. What we need is an interface.

1-6: USER INTERFACES

An interface is simply a component of a system specifically intended to allow the user to access the internal components of that system in a relatively easy fashion and without having to know specifically how everything is put together or how it works together. The easier it is for a user to access the system, the better the interface. Also, the more common the interface, the less effort and training it takes for a user to move from one system to another. For example, programs such as Microsoft Windows and the software applications written for it rely on a common interface. Saving and retrieving files in one Windows program is the same basic process as in all other Windows-based applications. The concept of the common interface is not limited to computer-based systems, however. Modern automobiles achieve an almost universal set of characteristics in their interface, which is why you don’t have to get a license to drive a particular car or type of car. Unless you try to drive something radically different in its interface, such as a semitrailer or school bus, you need just one license to drive. The interfaces are so similar that if you know how to drive one car, you know how to drive them all.

As you might expect, however, the world of DSSs does not yet enjoy the benefits of a universal or common interface. Even though it is a desirable goal, it is just not so easily accomplished. Remember, DSSs are not generic applications intended to be used by a variety of users in a variety of settings. On the contrary, a DSS is a domain-specific application that is normally designed for use by a small group. As such, the design and implementation of a common user interface is not easily attained. Typically, however, the limited user base for a particular DSS makes a common interface less important. Because the system is essentially “custom designed” for the user(s), the interface is also designed for the specific user or users of the system.

INTERFACE COMPONENTS

The DSS interface is responsible for all interaction and communication with the user(s). Given this level of responsibility, the interface must not only include software components (such as menus and command languages) and hardware components (such as multiple monitoring or input facilities), it must also deal with factors relating to human interaction, accessibility, ease of use, user skill level, error capture and reporting, and issues relating to documentation, among many others. Because of this breadth and depth of responsibility, DSS experts generally regard the interface as the single most important component in the system. Without a good user interface, the
power and functionality of the DSS remain inaccessible. Think of it this way: As far as the user is concerned, the interface and the DSS are one and the same. Poor interface equals poor system. In the past, lack of support for many DSS implementations has been linked to a poor user interface that served only to discourage use rather than facilitate it. Table 1-6 contains the basic components of a DSS interface and lists the factors that must be considered in its design.

Throughout this book we will explore the details of interface design as it relates to a specific DSS application context such as executive information systems, expert systems, or group decision support systems. For now, you can think of the DSS interface as having two components: the communication language and the presentation language.

The Communication Language
The communication language, or action language, component deals with activities associated with the user’s direct dialog with the DSS. The various modes of data entry are included in this component. Data can be entered into the DSS by conventional methods such as keyboard, mouse, trackball, or touchpad. More recently, however, expanded communication capabilities allow for a variety of not-so-common data entry mechanisms. Virtual reality devices such as biophysical input gloves, retinal scanners, and head position monitors are becoming common input methods. In addition, DSS applications using joysticks, voice recognition, and optical scanning devices are also being developed. Essentially, anything that can serve as potential input and can be captured as input is fair game for the communication language component of the interface. Figure 1-4 contains examples of common input devices that could be used with a typical DSS.

Probably the most common aspect of the communication language is the menu. Menus provide the user with an organized and intuitive method of selecting among the many functions, alternatives, commands, or outcomes available through the DSS. Properly organized and logically designed, menus can serve as guides for the inexperienced DSS user and as efficient vehicles of navigation for the DSS expert. Poorly designed menus, however, can render an otherwise powerful DSS unusable. As stated

---

### TABLE 1-6 General Functions of the DSS Interface

<table>
<thead>
<tr>
<th>Communication Language</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Allows for interaction with the DSS in a variety of dialog styles</td>
</tr>
<tr>
<td>• Identifies the form of input to enter requests into the DSS</td>
</tr>
<tr>
<td>• Provides support for communication among multiple DSS users</td>
</tr>
<tr>
<td>• Can be effected in a variety of formats including menu driven, question/answer, procedural command language, or natural command language</td>
</tr>
<tr>
<td>• Can capture and analyze previous dialogs so that future interactions can be improved</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Presentation Language</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Provides for the presentation of data in a variety of formats</td>
</tr>
<tr>
<td>• Allows for detailed report definition and generation by the DSS user</td>
</tr>
<tr>
<td>• Allows for the creation of forms, tables, and graphics for data output</td>
</tr>
<tr>
<td>• Can provide for multiple “windows,” or views, of the data to be available simultaneously</td>
</tr>
</tbody>
</table>
previously, we will explore the details of good interface design throughout the course of the book.

The Presentation Language
The presentation language component of the DSS interface is where all the action is. It is what the user actually sees, hears, or experiences during DSS use. Output devices such as printers, plotters, display monitors, audio monitors, and voice synthesizers are all part of this component. On-screen methods such as multiple windows, graphs, tables, charts, icons, messages, and audio triggers or alerts are also part of the presentation language. Just as the communication language component allows the user to transmit information and commands to the DSS, the presentation language component serves as the vehicle for the DSS to communicate with the user. These two components must be designed to work smoothly together if the DSS is to be considered user-friendly.

Research shows that the method by which information is presented to a user can dramatically affect its perceived value and how accurately it is interpreted. The presentation language component must be designed so that a wide variety of presentation methods can be either preselected as a preference or selected “on the fly” by the DSS user while viewing various stages of output. Various graphs, tables, charts, and combinations must be easily selectable by the user so that the output can be viewed from as many perspectives as possible. This flexibility ensures that the output is appropriate for the many different users of the DSS as well as for the range of problem contexts and decisions to be supported by the system. As with the communication language component, we will discuss issues of presentation in greater detail throughout the book within the context of specific DSS applications.
1-7: THE DSS USER

No DSS can be considered functional or complete without the user. Unlike many other computer-based information systems, in the DSS the user is as much a part of the system as is the hardware or software. The user is commonly defined as the person, or persons, responsible for providing a solution to the problem at hand or for making a decision within the context the DSS was designed to support. The DSS user is central to the entire life cycle of the system. Users may be involved in the conception of a DSS, in its logical and physical design, in its testing and implementation, and, of course, in its use.

USER ROLES

The user as defined in the previous paragraph, however, may not actually ever “use” the DSS. Alter (1980) classifies DSS users into one of five basic roles. He further classifies the various user roles into one of four basic patterns of use. Figure 1-5 shows the relationship between the defined user roles and their patterns of use.

In Alter’s classification, the user is defined as the person who communicates directly with the DSS regardless of method or intention. The decision maker is the person who ultimately makes the decision based, in whole or in part, upon the output from the DSS. An intermediary is a particular type of user who serves as a filter or interpreter of the output from a DSS. The intermediary may work closely with the decision maker to assist in interpreting the DSS output during the various stages of the decision-making process. The maintainer, or operator, is responsible for the daily operational aspects of the DSS,
such as keeping the system and its data up-to-date and in operational condition. Finally, the feeder provides data to the DSS but might not ever directly use the DSS as a decision support tool. The role of feeder may be filled by one or several people or groups that regularly generate data relevant to the problem context the DSS was designed to support. In some cases, all the user roles may be simultaneously occupied by a single person; in other cases, those roles may be managed by several different people or groups. Alter suggests that the fewer people involved in the various user roles, the simpler the DSS becomes to implement and use. As you can see, however, the realm of the DSS consists of many users who may never actually “use” the system to make a decision.

PATTERNS OF DSS USE

The patterns of use described by Alter help in understanding how various users may interact with the DSS. In the subscription mode the decision maker typically receives scheduled reports. Such reporting is generally preformatted and generated on a regularly scheduled basis. The reports are often generated automatically by the DSS and require no inquiry (other than the original one) on the part of the user. Moreover, these reports are often generated through consolidation of existing information submitted from several sources throughout the organization and are received either through off-line methods such as hard copy or, increasingly, through asynchronous electronic methods such as e-mail or file transfer. An example of subscription mode usage would be the periodic receipt of a budget variance report.

In the terminal mode, the decision maker interacts directly with the DSS in an online manner. In this mode, the user determines and provides the input, manipulates the models in the model base, and directly receives and interprets the output. The terminal mode user may use the received output to either make the final decision or determine the need for additional or more refined interaction with the DSS. An example of a typical terminal mode access would be the analysis of optimum stock portfolio combinations for a particular brokerage client.

A DSS can also be used in a clerk mode. Here, the decision maker uses the system directly but does not interact with the DSS in an online, real-time manner. Instead, the inputs, parameters, and requests are formed off-line and submitted via input coding forms or other electronic batch submission processes. While the decision maker awaits a response from the system, he or she can work on other activities. Situations in which a large number of users need access to a DSS but do not need online interactions are most compatible with clerk mode usage. Such systems can handle a large number of requests efficiently and can perform many complex analyses before returning the answers to the various clerk mode users. A common example of this type of system usage would be a DSS used to determine renewal rates for insurance customers or custom quotations for highly specialized service industries.

The fourth DSS usage category is the intermediary mode. Here the decision maker interacts with the DSS through one or more intermediary users. In this mode, the intermediaries are necessary due to the complex nature of either the analysis or parameter input process. In an extreme case, the DSS functions as the intermediary’s tool in the preparation of his or her final report to a decision maker. This mode of usage frees the decision maker from having to interact with the “automated beast” or to even know how the system works. A common example of this mode of use might be an organizational
CHAPTER 1 ♦ Introduction to Decision Support Systems

pricing structure that is predicated on multiple analyses by several divisions but ultimately must be consolidated by a human decision maker. The days of the intermediary usage mode are numbered, however. The windows of opportunity normally associated with typical managerial decisions are becoming ever smaller and more temporally demanding. For this reason, the time spent “interacting” with the technology through intermediaries may be prohibitive for many decisions. Tomorrow’s decision maker must have direct access to the support technology and must be skilled in its use. Furthermore, the designers of tomorrow’s decision support technologies must keep in mind the speed with which these technologies must be employed and how quickly answers must be obtained. For now, however, each of the four patterns of use has its place and, when properly applied, provides the necessary utility for the particular problem-solving context at hand.

1-8: CATEGORIES AND CLASSES OF DSSs

A variety of methods attempt to classify and categorize decision support systems. Methods based on the type of support offered by the DSS; decision situation; degree of user guidance or procedurality; orientation toward data, text, rules, or models; and focus on individual versus multiple decision makers are all used in classifying the myriad of DSSs in existence or under development. For the moment, however, it is enough to understand that the unique characteristics of a particular DSS classification may be important in determining the best approach to the design or implementation of a new system. We will explore these issues in greater depth in Chapters 14 and 15.

DATA-CENTRIC AND MODEL-CENTRIC DSSs

Building upon Figure 1-1 (page 8), Figure 1-6 contains a classification scheme for DSSs suggested by Alter. In this method, two primary support orientations for a DSS are used. The data-centric orientation focuses primarily on data retrieval and analysis sup-

![Figure 1-6](image-url)
port activities. The model-centric orientation includes activities such as simulation, maximizing or optimizing scenarios, and those DSS outputs that generate suggested actions based upon embedded rules or models. Within these two basic categories, a broad spectrum of DSS types can be positioned.

**FORMAL AND AD HOC SYSTEMS**

Donovan and Madnick (1977) proposed a method of DSS classification based upon the attributes of the problem-solving context. The formal, or institutional, DSS is designed to focus on periodic or recurring decisions within an organization. Such decision contexts require regular interaction with a DSS to ensure consistent and effective decision outcomes. Examples of this kind of system are found in periodic pricing scenarios typical of the petroleum industry or in dynamic decision contexts such as portfolio management or seasonal inventory control. The formal DSS tends to be stable in its design and evolves over a period of years into a highly refined, reliable support mechanism.

In contrast, the ad hoc DSS is designed to focus on a narrow problem context or set of decisions that is usually not recurring or easily anticipated. In these DSSs, the nature and immediacy of the decision situation drive the design and implementation considerations. A typical scenario for an ad hoc DSS might be the need to make decisions regarding a pending hostile takeover or corporate merger. Historically, the prohibitively high cost of developing ad hoc DSSs limited their availability and usefulness. However, the advent of DSS generators (generalizable software development environments that provide basic DSS components such as DBMS, MBMS, and knowledge management) made the ad hoc DSS a feasible, cost-effective method of providing high-quality decision support to “custom” application?

**DIRECTED VERSUS NONDIRECTED DSSs**

Silver (1991) proposed a method of classifying DSSs by the degree to which the system provides decisional guidance—that is, the manner in which a DSS guides its users in constructing and executing decision-making processes by providing assistance in choosing and using its operators. Silver defines operators as those elements of the DSS (menus, buttons, models, algorithms, tools) that can or must be manipulated by the user in the course of the decision-making process. This method of classification suggests that users may benefit from such guidance “imposed” upon the process by the DSS. Figure 1-7 illustrates a two-dimensional matrix of decisional guidance used to classify various decision support methods.

Silver’s classification initially categorizes a DSS by the type of guidance provided: mechanical versus decisional. Mechanical guidance is the more common form within a DSS and generally consists of assisting users with the “mechanics” of the operating system’s features such as menus, buttons, and commands. In contrast, decisional guidance assists the users in dealing with the various decision-making concepts relevant to the problem context. This category can be broken down further into guidance intended to assist in the structuring or execution of the decision-making processes. Finally, each decisional guidance subcategory can be classified by the form the guidance takes: suggestive or informational. The former proposes courses of action to the user. Examples of suggestive guidance would be a suggestion from the DSS regarding which operator to invoke in the next step or a recommended starting value as input to a
chosen algorithm. The latter provides users with information relevant to the situation at hand but does not indicate how the user should proceed based on that information. Silver contends that the type of guidance is likely to affect how the system is used as well as the decision outcomes made with the system.

### PROCEDURAL AND NONPROCEDURAL SYSTEMS

As discussed in Section 1–3, procedurality refers to the degree to which a user of a DSS can specify whatever information he or she wants from that DSS in whatever form he or she wants it. In a method of classification similar to Silver’s approach, the various DSSs can be positioned on a continuum according to their degree of procedurality. For example, a DSS using a highly procedural data retrieval language such as COBOL would require the user to provide detailed specifications regarding how the data are to be retrieved and precisely how each of the necessary computational subroutines is to be performed. The slightest deviation from a set of rigid syntactical rules such as a missing parenthesis or comma or a particular variable out of order would result in an unrecognizable command to the DSS, or worse, a command that, although recogniz-

---

**FIGURE 1-7 Silver’s Classification of Decisional Guidance**

<table>
<thead>
<tr>
<th>Form of Guidance</th>
<th>Suggestive Guidance</th>
<th>Informative Guidance</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Target of Guidance</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Recommended operator</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Set of recommended operators</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ordered list of recommended operators</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Set of operators not recommended</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Execution of the Process</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Recommended value</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Set of recommended values</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ordered list of recommended values</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Set of values not recommended</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Structuring the Process</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Description/analysis of operators</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Comparison of operators</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Map of relationships among operators</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Record of behavior in similar contexts</td>
<td></td>
<td></td>
</tr>
<tr>
<td>History of activity this session</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*Source: Reprinted by special permission from MIS Quarterly, Volume 15, Number 1, March 1991. Copyright 1991 by the Society for Information Management and the Management Information Systems Research Center at the University of Minnesota.*
able, may produce some highly unpredictable results. This system would be positioned at the extreme procedural end of the spectrum.

Systems using less procedural languages—written in typical fourth-generation languages such as Oracle or Smalltalk that employ a structured query language (SQL)—allow the user to use less of a procedural approach to the construction of commands or requests for information. These systems tend to fall near the middle of the procedural-ity spectrum. The nonprocedural command structure, though easier to understand and construct, nonetheless requires the user to follow a set of rules regarding sequence and syntax that is somewhat foreign to the way one naturally thinks and converses. The bottom line is that the user needs to learn a new vocabulary and grammar to use a command-oriented DSS effectively.

Recently, in an emerging trend toward natural language command languages, systems written in modern development environments such as Lotus Notes provide the user with a command syntax that closely resembles natural English language sentences. An extension of the concept of nonprocedurality, the natural language command processor accepts commands and instructions in the form of commonly structured English sentences that are sequenced and constructed at the convenience of the user rather than at the convenience of the command language. When we speak or write to each other, our sequence of words, punctuation, spelling, and grammar all convey meaning, and we can live with some fairly significant deviations from the rules without losing the essence of the meaning. In written form, if a comma is missing or a word is misspelled, we can usually still understand the message being conveyed. In a command-oriented language, regardless of procedurality, a missing comma or a command parameter out of order can render the message undecipherable by the command processor. The natural language DSS is more tolerant of these issues and can actually be designed to “learn” the user’s intentions over time. That is, the command processor can interpret the meaning of a new command within the context of the previous commands or requests made by the user. Additionally, natural language systems can be easily adapted to alternative forms of input, such as voice recognition or even visual pattern matching, that allow commands to be given to the DSS by simply speaking them or by making a specific motion or expression to a camera connected to the system. Despite the obvious advantages of a natural language command processor, however, this type of DSS is in its infancy. Technology is still a long way from providing users with machines they can interact with in a natural, humanlike manner. We will explore some of these issues in Chapters 9, 17, and 18.

HYPERTEXT SYSTEMS

Yet another method for classifying DSSs focuses on the technique used by the system to provide the necessary knowledge management for the problem context. One such technique is the document-centric hypertext system. Such systems support the decision process by keeping track of a large and often disparate knowledge base that is primarily document or text based. Activities including document creation, revision, searching, grouping, merging, indexing, and forwarding, among others, are typically found in a text-based DSS. Using the concept of hypertext, the text-based DSS can facilitate the user’s exploration of a particular flow of ideas through a group of documents. Various pieces of text within a document can be linked to related pieces of text in other documents so...
that a particular thread or train of thought can be explored. The most common example of this concept is found on the World Wide Web.

A user can begin at one Web site and by using hypertext links travel through literally millions of documents, or Web pages (see Figure 1-8), investigating a particular flow of ideas. At any time, the user can restructure the investigation to explore a new twist or turn or simply abandon the search and start over. This natural journey through related documents and pieces of text is driven entirely by the user and closely resembles the way human beings think and make associations among numerous concepts. Hypertext systems support the decision process by relating and remembering things for the user. Computers are far superior to humans at those tasks. The user is thus free to perform the decision process activities related to thinking and evaluating. These activities are still best performed by a human problem solver for most unstructured problem-solving contexts.

SPREADSHEET SYSTEMS

Another technique DSSs use for knowledge representation is the spreadsheet system. This simple, yet powerful, method of representing relationships through the use of a matrix of rows and columns allows the user not only to create and represent relationships in the form of mathematical or locational links but also to directly manipulate those linkages and see the end result. Application packages such as Microsoft Excel or Lotus 1–2–3 are powerful spreadsheet environments that can serve as a foundation for sophisticated and complex decision support systems. More recently, common develop-
ment environments such as Lotus Notes simply integrate these concepts of hypertext representation and spreadsheet systems. Using this approach, the DSS designer can incorporate spreadsheet representation of knowledge within a hypertext-constructed document so that the user can manipulate the data using a combination of the techniques as he or she sees fit.

INDIVIDUAL AND GROUP DSSs

Probably the most widely used method of classifying a DSS is based on its ability to provide support to a single decision maker or to a group of decision makers. In the modern business world, it is unrealistic to assume that decisions of consequence will be made by a lone decision maker. Rather, they are made by a collective or through the consensus of multiple decision makers. As such, the concept of a group decision support system (GDSS) suggests that the design, implementation, and use of such systems differ from that of DSSs intended for use by a single decision maker or problem solver. We will explore the similarities and differences between these two types of systems in Chapters 5 and 6.

1-9: CHAPTER SUMMARY

This chapter introduces you to the world of decision support and DSSs. By using the basic definitions and classification frameworks covered here you will be able to focus on the complexities and details of each type of DSS presented in the remaining chapters, and, in addition, you will be able to relate the use of a particular DSS to its appropriate problem context.

A DSS is truly a dynamic system that requires knowledge beyond that of the typical IS professional. The decision makers of tomorrow will rely on this technology to support their daily activities and to assist them in managing the ever-growing knowledge base necessary to make informed and effective decisions. Although this book cannot possibly serve as a single source of knowledge for computer-based decision support, it can and will provide you with a solid foundation for understanding, interacting with, and effectively using the decision support technologies of tomorrow.

Key Concepts

- Definition
  A decision support system (DSS) is a system under the control of one or more decision makers that assists in the activity of decision making by providing an organized set of tools intended to impose structure on portions of the decision-making situation and to improve the ultimate effectiveness of the decision outcome.

- Benefits/limitations of the DSS
  The manager should see the DSS as a valuable tool in the decision-making process rather than as a mechanism for the making of the decision itself.

- The components of a DSS
  The data management system
  The model management system
CHAPTER 1 ♦ Introduction to Decision Support Systems

The knowledge engine
The user interface
The user(s)

• Categories and classes of DSSs
  Data-centric DSS versus model-centric DSS
  Formal DSS versus ad hoc DSS
  Directed DSS versus nondirected DSS

Procedurality
Hypertext management systems
Spreadsheet systems
Individual versus group DSS

Questions for Review

1. What common characteristics of a decision support system relate to the decision-making process?
2. Why is a DSS a powerful tool for decision makers?
3. Is a “universal DSS” possible? Why or why not?
4. Why is the concept of procedurality important to the design and implementation of a DSS?
5. List and briefly describe the five basic components of a DSS.
6. What is a database? What are the possible sources of data collected in a database? Why is the ability to combine data records from multiple sources so important to DSS users?
7. What is a DBMS? Explain its two main responsibilities and how they contribute to the functionality of a DSS.
8. What is a model? How can a model base support decision makers in the problem-solving process?
9. Describe the two main responsibilities of the MBMS and the role of an MBMS in a decision support system.
10. What is reasoning? Why is reasoning important to the decision-making process?
11. What is a knowledge base? What is its role in a DSS?
12. How does knowledge get into the DSS? How can it be retrieved and organized into useful information?
13. What must be present in order for the user to be able to communicate well with the DSS?
14. Explain the roles of the various types of users of a DSS.
15. Describe and give an example of each pattern of DSS use.
16. Compare and contrast data-centric versus model-centric systems, formal versus ad hoc systems, and directed versus nondirected systems. What are the advantages and disadvantages of each type?
17. Compare the benefits and limitations of highly procedural languages and less or nonprocedural languages.
For Further Discussion

1. Analyze a DSS application in the market. Describe its main components and summarize its functionalities.

2. Observe the decision-making processes in an organization with which you are familiar. Discuss how a decision support system can help in the process. Decide which category of DSS can fit with the characteristics of those decision-making processes.

3. Compare a DBMS and an MBMS. What is common to both? What are the differences between them?

4. The admissions office at the University of Maryland needs a decision support system for evaluating applications. They need to build a database for the DSS. Specify the data sources and data that could be found for the admissions office to use.

5. Interview a member of an organization that is using a DSS. Specify the roles of the users and find their patterns of use.

6. DSS experts generally agree that the interface represents the single most important component in the system. Why is the user interface such an important component in a decision support system? Which kinds of interfaces are considered user-friendly?