Chapter 5

Designing and Evaluating DSS User Interfaces

INTRODUCTION

An effective user interface is one of the most critical components of any type of Decision Support System, but it is especially important for systems that will be used directly by managers. In the DSS literature, the user interface is sometimes called the dialogue component. Why is the user interface or dialogue component of a DSS so important? Research indicates that the easier it is to use a DSS, the greater the chance that managers will actually use the system. The user interface is what managers see and use when they work with a DSS.

Does the design of a user interface for a Decision Support System differ from the user interface for any other computerized system? What guidelines can improve DSS user interfaces? Managers should be the primary users of DSS, so a user interface should help managers interact with the system. When a user interface is complex or difficult to use, a staff person may need to assist a manager in using a DSS. A complex, difficult to use interface increases the operating costs of a specific DSS and probably limits its use. Can this negative outcome be avoided? If so, how? This chapter presents the basics of user interface design and the technologies that are used to create effective user interfaces. The focus of this chapter is on how display screens should look and how a DSS user interface should function. The chapter examines one specific design approach, a number of issues associated with building a DSS user interface, interface design elements, guidelines for user interface design, and factors that influence user interface success.

The general goal of user interface design is to develop screen layouts and interfaces that are intuitive, easy to use, and visually attractive (cf., Galitz, 1985). Both the intended users of a DSS and DSS analysts need to participate actively in designing and evaluating DSS user interfaces.

USER INTERFACES: AN OVERVIEW

Many DSS users have limited computing expertise. Most of these users do *not* want to learn a command language interface like Structured Query Language (SQL) that may be used by an decision support analyst or by a more technically oriented manager. According to Bennett (1986), for a nontechnical user, the design of an appropriate DSS user interface is the most important determinant of the success of a decision support implementation. So what is a user interface?

A user interface is what managers see and use when they interact with a DSS. More specifically, a user interface is the set of menus, icons, commands, graphical display formats, and/or other representations that are provided by a software program to allow a user to communicate with and use the program. A graphical user interface (abbreviated GUI which is pronounced "goo ee") provides a user a more or less "picture-oriented" way to interact with computing technology. GUIs remain controversial. Many people argue a GUI is the most user-friendly interface for a DSS. Some people disagree strongly with this conclusion. "User-friendly" is an evaluative term for one's subjective impression of a computerized system's user interface. It indicates that users judge the interface as easy to learn, understand, and use.

Also, a user interface refers to the hardware and software that create communication and interaction between a DSS user and the computer. The user interface includes responses and involves an exchange of graphic, acoustic, tactile, or other signs. User interface research is a subset of a field called human-computer interaction (HCI). HCI focuses on the study of people, computer technology, and the way each influences the other.

An effective user interface is important because the data and graphics displayed on a computer workstation screen provide a context for human interaction and cues for desired actions by a user. The user formulates a response to the context provided by the user interface and takes an action. Data then passes back to the computer through the interface.

A well-designed user interface can increase human processing speed, reduce errors, increase productivity, and create a sense of user control. The quality of a DSS interface, from the user's perspective, depends upon what the user sees or senses, what the user must know to understand what is seen or sensed, and what actions the user can and, in some cases, must take to obtain desired results.

To create a well-designed user interface, MIS professionals should work closely with potential users, try various design solutions, and provide users appropriate control over the functions of the system. This approach is often called User Centered Design (cf., Gulliksen, Lantz, and Boivie, 1999). Both groups of design participants need to be familiar with the following important issues and topics related to building and evaluating a user interface:

1. *User interface style* – Is the style or combination of styles appropriate? What styles are used in the user interface?
2. *Screen design and layout* – What design approach should be used? Is the design easy to understand and attractive? Is the design symmetric and balanced?
3. *Use of colors, lines and graphics* – Are colors used appropriately? Do graphics improve the design or distract the user?

4. *Information density* – Is too much information presented on a screen? Can users control the information density?
5. *Use of icons and symbols* – Are icons understandable?
6. *Choice of input and output devices* – Do devices fit the task?
7. *The Human-Software interaction sequence* – Is the interaction developed by the software logical and intuitive? Do people respond predictably to the interaction sequence?

Managers and DSS analysts should focus on these seven design issues when they evaluate a DSS prototype or the proposed screens for a DSS. A systematic evaluation of a DSS user interface can substantially improve its usefulness and increase how much it will be used. Let's examine some of these issues in more detail.

USER INTERFACE STYLES

The user interface determines how information is entered and displayed. The interface also determines the ease and simplicity of learning and using the system. There are four general structures or interface styles that can be used to control interactions with computerized information systems. These styles are: 1) command-line interfaces; 2) menu interfaces; 3) point-and-click graphical interfaces; and 4) question-and-answer interfaces. Each style can be used in creating DSS user interfaces. The styles can often be combined usefully in a single application or set of related applications (see Galitz [1985] ; Shneiderman [1992] ; and Turban [1995]). When building a user interface, a designer should try to provide multiple ways to perform the same task. For example, a design may include a command-line interface, pull-down menus for commands, and keyboard command equivalents. Many input devices, including keyboard, mouse, touch pad, and voice inputs, can be used to manipulate these four general interface styles.

Command-line Interfaces

Command-line interfaces are the oldest form of computer control. They originated when each command to a program was entered on a punched card. A command-driven interface still dominates some operating systems, including MS-DOS, UNIX, and Linux. In a DSS with a command-language style interface, a user enters a command such as "run" or "plot." Many commands are composed of a verb-noun combination (for example "plot sales"). Command-line interfaces require a user to enter a command telling the system what to do next. It is the user's responsibility to know what commands are available and how to phrase those commands with their parameters. Such interfaces can be quite powerful, giving their users detailed control over system operation, but there is a significant cost in terms of increased training. Command interfaces are hard to learn. Managers must attend training workshops and read documentation. Most people never learn more than a fraction of the commands in any command language and make frequent mistakes in command entry. While command entry mistakes can usually be corrected, they create a cost for

users and companies in terms of productive time lost, and frequent mistakes can make users feel frustrated and even incompetent.

Menu Interfaces

In a menu interaction, a user selects from a list of possible choices the task or function to be performed. The ordered list of functions or tasks is called a menu. The user makes a choice among items by manipulating an input device or entering a menu item number. Menus should appear in a logical, hierarchical order, starting with a main menu and going to subordinate or submenus. Menus can become tedious and time-consuming when complex situations are analyzed, since it may take several menus to use a system and the user must shift back and forth among the menus. A pull-down menu is a submenu that appears as a superimposed drop-down menu on a screen, usually after an entry has been made in a high-level menu. A tool bar with graphical icons can also serve as a menu.

Menus are often effective because they rely on recognition rather than recall. Working with menus reminds users of available options. The menu designer must consider the conflicting needs of both experienced and inexperienced users.

Graphical Interfaces

A graphical user interface (GUI) is an interface system in which users have direct control of visible objects. Users point and click to initiate actions rather than enter complex commands. Two well-known GUI are the Windows 95/98/2000/XP operating systems and the Macintosh OS. The major GUI elements are windows, icons, pull-down menus, and dialog boxes. A window is an area of the computer screen that behaves as if it was an independent computer terminal. Icons are small pictures that represent windows or actions. Some of the icons frequently used in Microsoft applications are shown in Figure 5.1. Clicking on an icon initiates opening a window or running a command. In the graphical or object manipulation interface style, the user directly manipulates objects represented as symbols called icons.

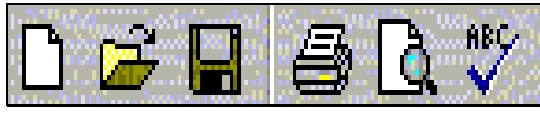

Figure 5.1 Examples of Icons.

User interfaces can be enriched with the use of multimedia and hypermedia technologies. Multimedia refers to many media, including graphic materials, audio, and images, including motion pictures and animation. Hypermedia describes documents that contain several types of media linked in documents. The World Wide Web is an example of a hypermedia delivery system. Web

documents can include explicit internal and external links, multimedia content, and interactivity with databases.

Question-and-Answer Interfaces

A question-and-answer interface dialogue begins with the computer asking the user a question. The user answers the question with a phrase or a sentence. A dialogue then occurs between the computer and user. The computer's questions are a function of prior responses of the user and the processing needs of the application. A related interface style is called form interaction; in form interaction style, the user enters data or commands into designated spaces (fields) in a form. The headings of the form serve as a prompt for the desired input. A human-computer interaction that is similar to a human-human dialog is referred to as natural language dialogue. The major limitation of using natural language responses is the inability of the computer to really understand unstructured or unanticipated natural language. The programmer must anticipate user answers and program responses.

The following example shows a simple question and answer dialogue:

>*dss*	What is your name?
>*user*	Daniel Power
>*dss*	What is your age?
>*user*	51 years old
>*dss*	Please enter a number
>*user*	51

A question and answer dialogue is one of the oldest types of interfaces; it is not used as frequently today in building DSS, but it may be revived by improvements in speech recognition technologies.

Another new type of interface is called a three-dimensional or virtual reality (VR) interface. It is being used in a number of research settings. With a VR interface, a user interacts with a computer-generated environment. A user wears a headset and hand-position sensor to interact with the decision support simulation. A user can walk around, grasp, and move objects, and, in general, alter the environment. A VR interface may become a viable DSS user interface in the future, but for the next few years managers and DSS analysts should focus on the interfaces discussed in preceding paragraphs. In most DSS more than one interface style is implemented.

ROMC DESIGN APPROACH

Sprague and Carlson (1982) presented an approach for designing DSS and especially the user interface called ROMC. Their approach has four user-oriented entities: 1) Representations for conveying information to the user, 2) Operations for manipulating data displayed as representations, 3) aids for a user's Memory, and 4) aids for helping users Control a DSS.

This section describes the four components of Sprague and Carlson's approach and provides contemporary examples of each component. ROMC was

intended as a process-independent approach for identifying the necessary capabilities of a DSS. It can also serve as a framework for creating screen designs and for building the user interface of a DSS. DSS analysts can improve screen design and layout by focusing on these four components as user interface design elements.

Representations

In a DSS, decision-making activities take place in the context of a conceptualization of the information used in the activity. The conceptualization may be an icon, a chart, a map, a text document, a form, a spreadsheet, a picture, a table of numbers, or an equation. The conceptualization is a physical representation that helps a decision maker communicate about the decision situation with another person.

Representations provide a context in which users can interpret DSS outputs and select DSS operations. Representations also can be used to supply parameters for DSS operations. For example, a point selected on a graph or a map can be linked to a data value, a document, or a database query. Also, prioritizing a list of employees may be the primary input for a personnel-scheduling DSS. Managers and DSS analysts need to evaluate and choose appropriate representations.

Operations

Operations are specific tasks that a decision maker can perform with a DSS. For example, a DSS may have operators to gather data, generate a report, retrieve alternatives, rate alternatives, add alternatives, etc. Note that an operation may be used in more than one activity and that there is usually no prespecified ordering of operations. Analysts need to decide how operations will be controlled from the user interface. Will menus be used? Icons? What names will be used for operations?

Memory Aids

Several types of memory aids should be provided in a DSS user interface to support the use of representations and operations. A symbolic link to a data warehouse is a memory aid for decision makers. Triggers or rules remind a decision maker that certain operations may need to be performed. A user profile or data filters may make operation of the DSS easier. User-established links or command sequences can make a specific DSS easier for that user to manipulate.

A trigger may invoke an operation automatically or remind the DSS user to invoke the operation. A profile can store initial defaults for using the DSS. Users' logs of actions taken and operations invoked are also memory aids, especially if the user can back up and undo or replay actions. DSS analysts should identify needs for memory aids and decide how reminders will be displayed. The help system is an important memory aid that must be designed as part of the user interface.

Control Aids

DSS control aids are intended to help decision makers use representations, operations and memory aids. Control aids help decision makers direct the use of the DSS. One type of control aid focuses on the standard conventions for user-system interaction, which are enforced across representations and operations. This type of control aid uniformly displays menus or defines guidelines for the design and behavior of icons. Some operations are more system-oriented than decision-process-oriented and these operations are also control aids. Edit, delete, and save operations are generic control operations and hence they are also control aids for the DSS. The tools used to create the user interface constrain the control aids. User interface design guidelines should also standardize the "look and feel" of the user interface.

BUILDING THE DSS USER INTERFACE

The ROMC framework can be a useful tool for designing the DSS user interface. Also, the ROMC specification of elements, along with screen layouts, can aid in implementing the actual DSS user interface. Every DSS will have a specific set of representations, operations, memory aids, and control aids. The generality and usefulness of a DSS will depend on the skill of the designers in selecting design elements.

Flow-charting the existing or a desired decision process can help develop the ROMC framework. A decision-process flowchart should focus on the inputs, operations, and outputs of each decision task. The resulting DSS design will likely follow the flowchart and its sequencing of tasks. The resulting representations may be effective, but the operations and control aids developed from this approach may provide limited flexibility to the decision maker. Creating prototypes of the DSS screens early in the analysis process, and then eliciting input from potential users, can reduce the problem of limited flexibility in the operations of a DSS.

Screen designs and layouts should be aesthetically pleasing. The design does not need to be "artistic," but it should not create a negative impression. Managers and designers should evaluate a DSS user interface in terms of balance, symmetry, proportion, and arrangement. Balance means the design elements are equally weighted on the screen. Symmetry refers to correspondence in size and shape of the design elements. Proportion is a harmonious relation among the parts. Arrangement is the ordering of elements. A balanced, symmetric screen design is the easiest screen layout to create and it is generally pleasing. Working with unbalanced and asymmetric screen designs is much more difficult for most of us. Figure 5.2 provides an example of a simple screen design. Does the design appear balanced and symmetric?

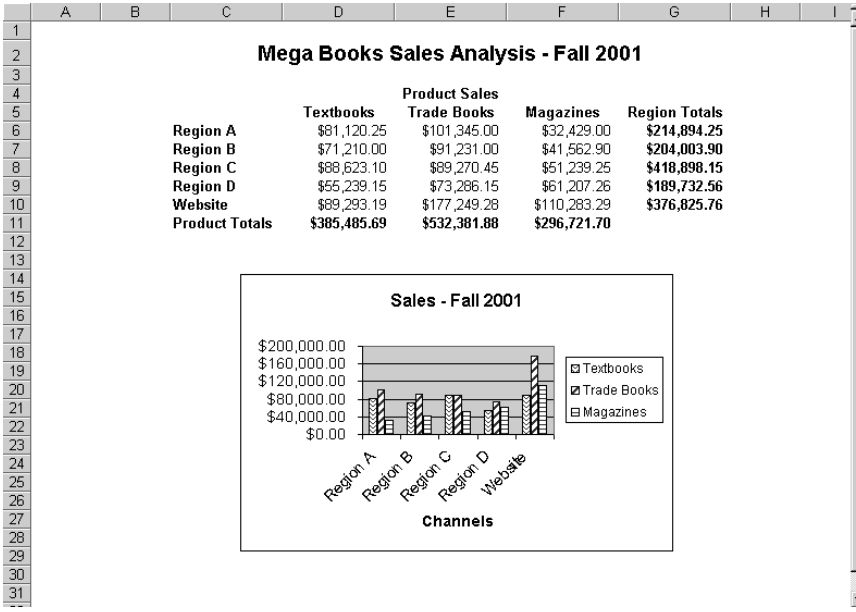

Figure 5.2 An Example of a Simple, Balanced Screen Design.

Keen and Gambino (in Bennett, 1983, p. 168) provide the following suggestions for building a DSS user interface. They believe rapid prototyping and adaptive design is essential for building the user interface; they argue that a DSS analyst, programmer, or consultant who is building a DSS must do the following:

a. *Get started*. A DSS application does not usually come packaged with neat specifications. Start with an initial user interface design. It provides a means of learning from and responding to the user.
b. *Respond quickly*. A DSS user interface must evolve rapidly, and designers must learn quickly. The design structure and programming techniques must facilitate evolution and learning.
c. *Pay close attention to user-system interfaces and outputs.* A DSS is "a set of relatively simple components that must fit together to permit complex, varied, and idiosyncratic problem solving". A DSS analyst needs to get a very detailed understanding of the task to be supported and of the people who carry out the task.

According to Keen and Gambino, the "natural sequence and order of priority" in developing any type of DSS is the following four steps:

1. Design the user interface and dialog.
2. Design commands and operations in terms of the users' processes and concepts.
3. Define what the user does and sees when a command is invoked.
4. Work backward to create the program logic and data management.

Professor Mark Silver (1991) proposed an alternative design approach. He suggests the following steps as appropriate for developing a DSS user interface:

1. Determine who your user is.
2. Determine what the user will do with the system. What are the specific tasks?
3. Determine what sequence of steps the user must follow to accomplish a task.
4. Diagram the steps in item 3 and the decision tree involved. Review them with the user.
5. Determine which of these steps require interaction with the system.
6. Determine information and decision requirements for each interaction (both system and user).
7. Select the categories of dialogue (menus, prompts, forms, etc.).
8. Diagram the flow of dialogue, showing all decisions and their information requirements. Review these with the user.
9. Design screens.
10. Try it, analyze it, simplify it, change it, try it . . .
11. Update the decision diagrams.
12. Bulletproof the dialogue by asking what happens if the user does something unexpected?

While Silver's list focuses on a broad set of steps, steps 4 to 11 are an iterative design process. Even if a DSS analyst plans to develop a DSS user interface using rapid prototyping, it is important to understand who the DSS user is, what the system will be used for, and what sequence of steps a user will follow. A designer may be able to skip over some of the formal diagramming steps, such as steps 4, 8, and 11, in favor of creating a prototype, but some understanding of the task should be formalized and documented.

In general, a DSS analyst should complete the design of the user interface prior to building a database and implementing the design. During construction of a DSS changes will be made, but the interface design forces an analyst to deal with many practical issues. A DSS analyst can use a number of tools to assist in interface design including screen mock-ups, transition diagrams, and menu trees. When possible the software that implements the user interface should be decoupled from the DSS data, model and communication components.

COMMENTS ON DESIGN ELEMENTS

Graphics, including charts, enable the presentation of information in a way that can clearly show the meaning of data and permits users to visualize relationships. The importance of using charts and graphs in communicating numeric data has been recognized for many years. Since the mid-1970s, computer graphics have been used to aid in management decision making. Graphics help managers "visualize" data, relationships, and variances. Common types of computer graphs and charts include time-series charts, bar and pie charts, scatter diagrams, maps, hierarchy charts, and flow charts. Managers, business analysts, and corporate staff use computer-generated graphics in reports, presentations, performance tracking, scheduling, control, planning, modeling, and design. It is important to use graphics in the DSS user interface,

especially in data displays. An end user tool like Excel has a wizard that helps create charts. Some of the charts available in Excel are shown in Figure 5.3.

Figure 5.3 Some Excel Chart Types.

Let's summarize some guidelines for graphical displays. Communicate only one major message on each chart or screen. Use an action heading in appropriate sized fonts for screen and chart headings. A line chart is appropriate for displaying time-related information, but analysts need to be careful to use appropriate labels and to avoid adding dissimilar quantities. Bar charts are more appropriate for comparing individual data values. Pie charts help show how the whole breaks down into component parts. You should limit the number of components or pieces in a pie chart to five or fewer.

Color is often recommended as a means of enhancing a user-interface design. Appropriate use of color can enhance the aesthetics of an interface for most people. Color can call attention to extreme or exceptional data values, help users differentiate among items on a chart, and convey information quickly. For example, research indicates blue creates a sense of trust, green means "go" or "all clear;" and red indicates danger. In general, the following guidelines related to the use of color are appropriate for DSS user interfaces:

1. Do not allow color to be the only way your system conveys any information. Supplement the use of color with other cues that can be used by people who cannot perceive the color difference. Include numerical values in addition to color codes, provide cross-hatching on top of color, or make sure that the colors chosen are perceived as substantially lighter or darker than each other.
2. Where computer hardware and software permit, allow the user to customize an application's use of color. Changing colors can compensate for some people's color vision deficiencies. In some systems, colors that cover an area, such as a region on a map, can be replaced by monochrome patterns such as dots, stripes, and cross-hatching.
3. Use light pastel colors in screen designs. They create fewer annoying reflections than do dark colors. This is especially true in an office environment

with fluorescent lights. As a result, try to use light colors to cover large areas of the screen. Save darker colors for smaller "spot" usage.

The use of color in a DSS user interface can create accessibility issues for some people with vision impairments, but if multiple cues are used, the system remains accessible and added benefits can be gained from the appropriate use of color cues.

How people interact with a system (human-software interaction sequence) is also an important design issue. One issue, unique to DSS design and often inadequately considered, is the type and amount of guidance, called decisional guidance, that a DSS provides its users in the decision-making process (cf., Silver, 1991). Decisional guidance provided by a system can be unintended or inadvertent. For example, users tend to select the first or last items from menus. Putting a frequently used capability in the middle of a menu may not be planned, but it may cause problems. Decisional guidance can also be planned and deliberate. Designers can intentionally build guidance mechanisms into a system after determining that a particular decision approach or process is better than what many users would come upon by chance. This type of planned process guidance is distinct from the typical on-line help facility, which focuses on guidance in the mechanical aspects of operating the system. On-line help assumes that the user has already decided what to do, but does not know how to do it. Process guidance assumes the user needs direction in using the system for decision support.

Some DSS give the developer more of an opportunity to provide decisional guidance than do others. A DSS that does not provide for many discretionary user judgments during its use cannot benefit greatly from such a guidance facility. A DSS that lets users choose among several decision methods, alternative models, ways to cross-tabulate a set of data, forecasting techniques, or even alternative sequences of activities, does provide an opportunity to provide decisional guidance.

GUIDELINES FOR DIALOG AND USER INTERFACE DESIGN

This section is based on Ben Shneiderman's (1992) research and writings. He has developed some underlying principles of design that he argues are applicable in most interactive systems. The following underlying principles of interface design are derived heuristically from experience.

Strive for consistency. This principle is the most frequently violated one, and yet is the easiest one to apply. Consistent sequences of actions should be required in similar situations; identical terminology should be used in prompts, menus, and help screens; and consistent commands should be used in a DSS.

Provide shortcuts for frequent users. As the frequency of use increases, so does a user's desire to reduce the number of interactions and to increase the pace of interaction. Frequent knowledgeable users appreciate abbreviations, special keys, hidden commands, and macro facilities. Shorter response times and faster display rates are other attractions for frequent users. A system must respond to the differing needs of its users.

Provide informative feedback. For every user action, there should be some system feedback. For frequent and minor actions, the response can be modest, whereas for infrequent and major actions, the response should be more substantial.

Design dialogs to create closure. Sequences of actions should be organized into groups with a beginning, middle, and end. The informative feedback at the completion of a group of actions gives the user the satisfaction of accomplishment, a sense of relief, the signal to drop contingency plans and options from their minds, and an indication that the way is clear to prepare for the next group of actions. The "power" of a dialog and commands should be appropriate to the capabilities of the users. "Power" is a measure of the amount of work accomplished by a given instruction to a system (cf., Galitz, 1985).

Provide simple error recovery. Design the system so the user cannot make a serious error. If an error is made, the system should detect the error and offer simple, comprehensible mechanisms for handling it. A user should not have to repeat actions, but rather should need to repair only the faulty part. Errors should leave the DSS unchanged, or the system should give instructions about restoring the prior state.

Permit easy reversal of actions. A user's actions should be reversible. Following this guideline relieves anxiety, since the user knows that errors can be undone, and encourages exploration of unfamiliar options. The amount of reversibility may be limited to the most recent action or to only data entries, or it may be essentially unlimited.

Support internal locus of control. Experienced users want to feel that they are in charge of the system and that the system responds to their actions. Surprising system actions, tedious sequences of data entries, incapacity or difficulty in obtaining necessary information, and the inability to produce the action desired all build anxiety and dissatisfaction.

Reduce information load. Information load is a measure of the degree to which a person's memory is used to process information on a display screen. It is a function of the task being performed, a person's familiarity with the task, and the design of the user interface itself. The limitation of human-information processing in short-term memory requires that displays be kept simple and that sufficient training time be allotted for learning commands and sequences of actions. Where appropriate, on-line access to command-syntax forms, abbreviations, codes, and other information should be provided. Designers can reduce information load by providing graphic rather than alphanumeric displays, formatting displays to correspond to users' immediate information requirements, using words that are easy to understand, and providing simple dialogues (cf., Galitz, 1985, p. 21).

FACTORS INFLUENCING USER INTERFACE DESIGN SUCCESS

According to Larson (1982), user interface design success is influenced by 11 factors. Some of his factors overlap with Shneiderman's (1992) guidelines. System factors include uniformity of the commands and interface, adaptability, execution time, system versatility, and quality of help provided. Human factors

include the learning time for the DSS, ease of recall, errors made by users, concentration required, fatigue from using the system, and the "fun" the user has while using the system. Larson and others have explained each of these factors that determine, or at least affect, DSS user interface design success.

Uniformity of commands and interface. Are the commands of the DSS identical to equivalent commands of other systems? If they want to, people can learn any idiosyncratic or esoteric interface; what is difficult is learning and remembering two, three, or more interfaces and switching frequently among them. A DSS developer must be aware of other systems used by the target managers and then strive for consistency in the user interface.

Adaptability. Does the system adjust to the end user's level of competence as he or she becomes more experienced? Does it tailor itself to the habits and styles of different users? Does the DSS provide shortcuts for frequent users? It may be difficult or impractical for a DSS to be "self-tailoring" in this sense. It is easier, and may be sufficient, to let an experienced user select an "expert user" mode in which prompts are minimized. In a graphical user-interface environment, it is helpful to provide keyboard equivalents for commonly used mouse-and-menu commands, as some users prefer using a mouse and others prefer using keystrokes.

Execution time. How long does it take the user to perform his or her decision support task? Decision task time is influenced by the choices made for software, hardware, and the user interface design. In general, faster execution times increase design success. User interface design can minimize time wasted by users.

Versatility. Can the DSS be used to perform a variety of tasks? A DSS must be versatile enough to accommodate the full range of tasks that need to be performed by a decision maker who wants to use it. Once a DSS becomes widely used, tasks related to its original purpose—but distinct from that purpose—may be performed by users. This is especially the case with data-driven and document-driven DSS. While it is possible these new tasks will require additional development work, it should be possible to incorporate them into the existing user interface.

Quality of Help provided. Does the system provide help when the user has trouble? On-line help facilities are becoming the norm for DSS. Many development tools make it easy to incorporate on-line help into a system. Wherever possible, help should be context-sensitive. The Help facility should recognize what the user is trying to do, or at least what screen the user is looking at, and provide help that is tailored as closely as possible to the current need.

Learning time for the DSS. How long does it take a novice to learn the system? A design that provides for rapid learning must take into account what the user knows and how the user's mind fits that knowledge together.

Ease of recall. How easy is it for an end user to recall how to use the system after he or she has not used it for some time? This is a more important factor for DSS than for transaction processing systems because managers often return to a DSS after a long interval of non-use. For example, some budgeting support systems are intended to help with decisions that recur predictably on an annual basis. Eleven months might elapse from a manager's last use of the system until

his or her next use of it. A user interface that facilitates recall will reduce the time it takes to get back up to speed each year.

Errors. How many errors does a DSS user make, and how serious are those errors? Does the DSS provide simple error recovery? The most serious errors can lead to wrong decisions. Following closely behind in significance are errors that corrupt a corporate database. Then come errors that bring down or "crash" the computer, followed finally by errors that waste the user's time but have no other bad effects. Fortunately, most user errors will be in the last category. Understanding a user's usual decision-making process can help minimize errors.

Concentration required. How many things must a user keep in mind while using the DSS? Most people have difficulty keeping more than six or seven active facts in mind at any one time. One way to reduce the memory load is to label screens and output with informative labels like: "Profit Projections," "Sales Growth," and "Model 47 shipments start 4/94."

Fatigue. How quickly does the user tire while using the system? What is the information load created by the DSS? Physical fatigue is seldom a factor with DSS because usage frequency is not high enough to lead to such a problem. However, mental fatigue can occur. DSS analysts should minimize mental fatigue by examining the concentration required and keeping it within the capabilities of the intended users.

Fun. Does the manager enjoy using the DSS? This does not mean "funny" error messages or jokes on the screen. Such "humor" grows stale quickly. It means designers should keep users informed about what the system is doing, warn them of time-consuming operations, provide progress displays, and generally try to minimize frustrations that come from using an uncooperative system. Simulations and "what if" analyses can also help make a DSS "fun."

CONCLUSIONS AND COMMENTARY

Evaluating a DSS user interface is as important as designing it. Managers are the evaluators, and hence their evaluation during the development of a DSS interface design influences its success. Much guidance has been written about computer software user interfaces—because this topic is important—and much more can be written—because there remains so much to learn. The user interface is a critical component of all DSS. It facilitates communication between a DSS and its users. The intended user can, however, create unique design and development problems.

This chapter briefly examined and evaluated knowledge about user interface design. In general, it seems that managers are rarely comfortable with command-line or simple menu interfaces. It appears that a sophisticated graphical interface can increase a manager's use of computerized decision aids and Decision Support Software. DSS users seem to prefer easy to use, functional interfaces that use meaningful design conventions and standards. "Cute" user interfaces with funny graphics lose their appeal quickly and complex interfaces increase training costs and increase the burden to recall commands and conventions for the user. The use of graphics and other visual information displays seems to appeal to many managers. GUI seems here to stay.

A good DSS user interface can be built using a number of development environments. A Web-based "thin client" interface can be as powerful and as easy to use as a more traditional client/server interface with a "thick client" installed on each user's computer. But, in any development environment, extensive time and energy need to be spent on designing and evaluating the interface. A good interface needs to be planned and designed.

Finally, it is advantageous to use a checklist for evaluating a prototype or proposed DSS interface. The DSS User-Centered Design Guidelines in Appendix II provide a starting point for creating a more specific list. Determine that the DSS content is easy to understand, that the DSS is robust and resilient, that it has an appropriate orientation and useful navigation tools, and then check the impact on productivity, the impact on data integrity, and the control capabilities for users. Overall, if DSS analysts improve user interfaces, then the usefulness of a new DSS and its value to decision makers should increase.