The Application System/400 help facility—design philosophy and considerations

by D. A. Charland

The design of the Application System/400™ (AS/400™) system help facility was based on the philosophy that users must be able to quickly access the specific information needed to complete their immediate task. This philosophy, reinforced by experience with help information on earlier IBM systems, resulted in a modular help facility that provides two major types of assistance discussed in this paper. Contextual information based on cursor position is provided for each panel. This contextual help is supplemented by an index of how-to-do-it and what-it-means information that can be searched by users in their own words.

Ideally, a user's interaction with a computer system should be so natural and intuitive that no help is needed. For a multiuser system like the Application System/400™ (AS/400™), which provides an extensive array of functions and applications and is used by persons whose backgrounds range from a novice with virtually no computer knowledge to an expert programmer, the ideal is not likely to be achieved for all users. There will always be some who need more assistance than others. Furthermore, a person who is expert at using one part of the system or application may be a novice at using another part.

A useful technique for accommodating these differences in experience and knowledge is to supplement the primary user interface with a help facility that provides additional information at the user's request. By providing the information only when requested, a help facility does not interfere with a user's normal interaction with the computer. What is less well understood, however, is the type and amount of help information to provide and the best ways of giving users access to this information.

Design philosophy

Testing and interviews of users of the AS/400 system's predecessors, the System/36 and System/38, indicated that when users ask for help, their primary—and frequently only—purpose is to get the information they need to complete their current task. The user needs sufficient information, but quickly becomes frustrated if presented with screen after screen of text. Each person's view of what is sufficient information depends on that person's experience and knowledge, which varies not only for different users but also for the same user over time as particular aspects of the computer system become more familiar.

The design of the AS/400 help facility began, therefore, on the simple premise that the facility will succeed or fail, depending on the efficiency with which it provides specific information needed by users to complete their current task. The on-line help is not intended to provide all information about the system, because an extensive library of printed manuals is available. Nor is it intended to teach users about the system (although users could certainly learn more about the system from the help facility). Comprehensive on-line education is available to provide users with an understanding of system concepts and facilities.

The information module concept. Breaking up information into small chunks to make it easier to grasp is a well-known technique.[1] As a result of experience with previous on-line help facilities, both as a user and as a developer, the author and several associates became convinced that using a similar chunking technique for on-line help would allow users to access the specific information they need without having to hunt through masses of extraneous information. We proposed, therefore, a modular help design in which the on-line information is split up into small, separately accessible building blocks called *information modules*.[2] Conceptually, the help facility is more like an information database than a sequential book.

In this modular help concept, each information module consists of the smallest unit of information a user would want to access separately. Examples of information modules are:

- The description of one option on a menu
- The description of one parameter of a command
- The description of how to perform one task
- The explanation of one technical term

The information modules can be accessed individually or put together like building blocks to form a larger set of information, as needed to satisfy the information request of a user. Consider, for example, a panel of information presented on a screen as shown in Figure 1. Typically, a panel contains a set of distinct items with which a user interacts, such as menu options that the user can select or entry fields in which the user can type information. For each item, there is a corresponding information module that users can access directly without having to hunt through the information about other items. Users

Figure 1 Information modules accessed separately or as a set

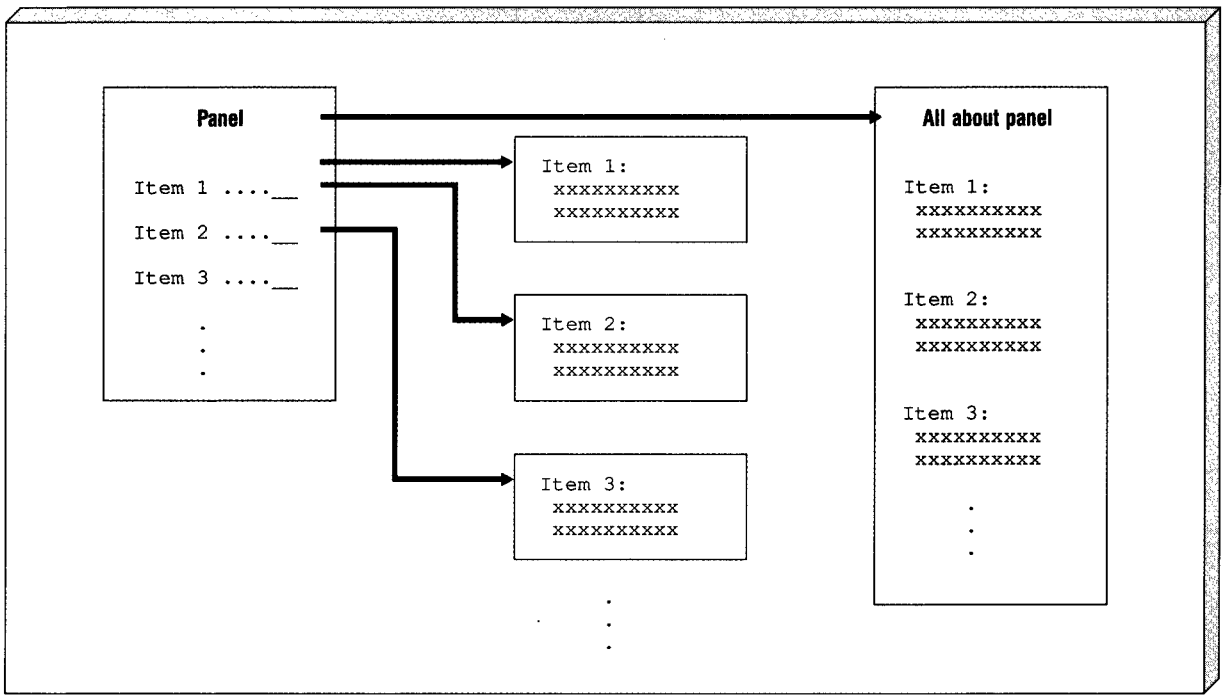
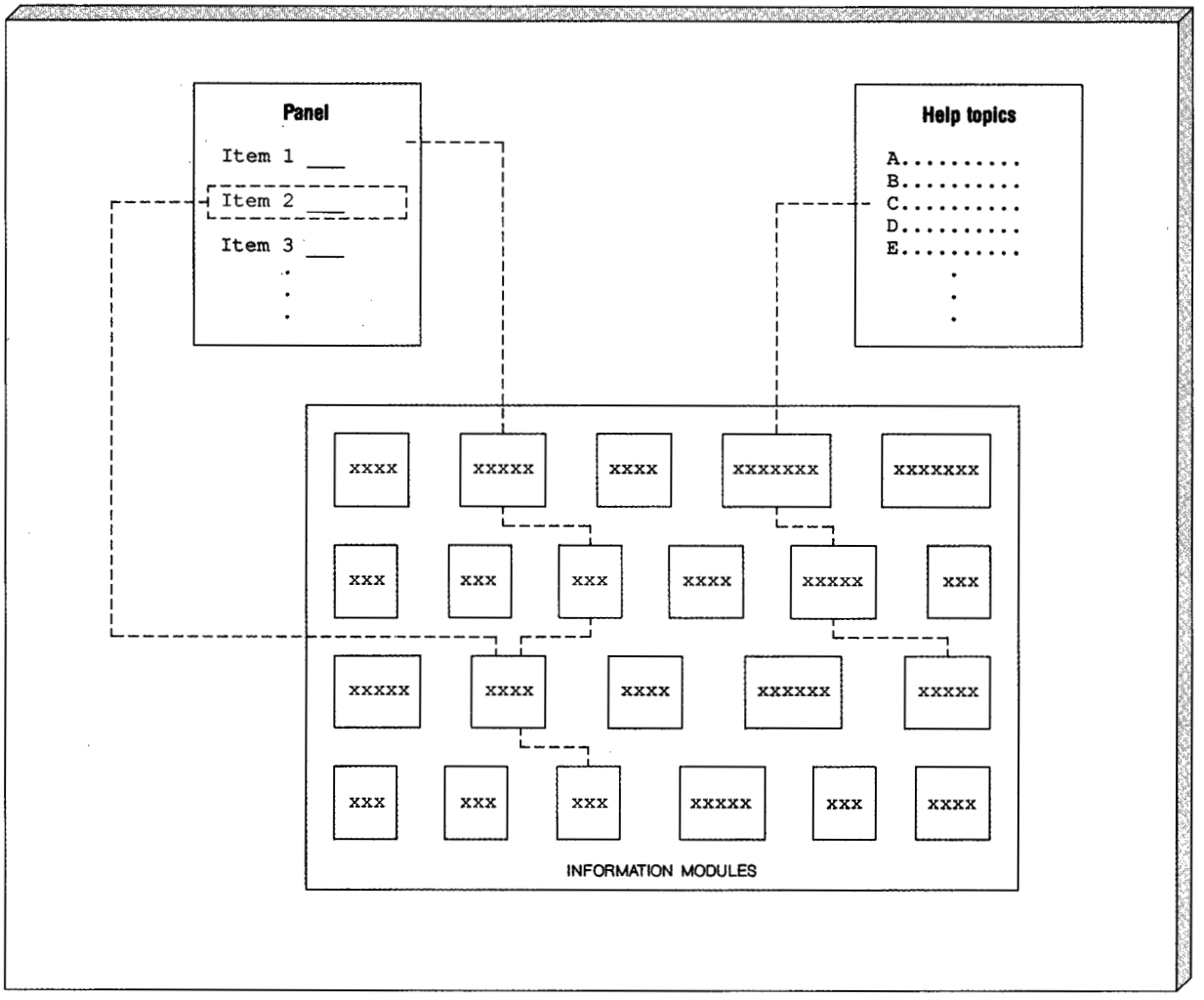


Figure 2 Database of information modules accessible in different ways and combinations



should also be able to see the full set of information if they so desire. This capability is provided by linking the individual information modules together, as indicated in Figure 2, so that they appear to be continuous lines of text.

Information modules may contain information other than that related to specific panels. The modules may be linked together in different combinations and accessed in different ways, as indicated in Figure 2. The same information module may be used in several different contexts, even though each information module is created and stored in the system only once.

Structure of the help facility. Users ask for help because they are having difficulty using the computer. Therefore, it is critical that all users be able to grasp quickly the structure and use of the help facility, so that the facility itself does not become an impediment. The help facility must be simple, even at the expense of eliminating some function. At the same time, the access methods must be sufficiently flexible to accommodate different user perceptions of what they want to do on the system.

Contextual help for panels. Following the premise that users want information related to their current task, we assume that a user who requests help by

pressing a help key wants, at least initially, information about the panel currently being used. The specific information provided is determined by the cursor position on the panel. If the cursor is in a clearly definable area of the panel, such as a menu option, an entry field, or a list column, *field help* is provided for that specific area, as indicated by path A in the upper half of Figure 3. If the cursor is in a less specific area, such as a panel title or a blank line, *extended help* is provided about the use of the panel as a whole, as shown by path B. Extended help consists of a general description of how to use the panel, followed by all of the field help, which is presented as continuous scrollable lines of text. After looking at the help for a specific field, some users might want more general information about the panel. Therefore, a function key is activated on all field help panels to allow the user to go directly to the beginning of extended help.

Some users may need supplementary information that describes the "big picture" that comprises multiple panels.

Index search. The contextual help provides the immediate assistance for a user to successfully interact with each panel, but it is not sufficient to satisfy the information needs of all users. Some users may need supplementary information that describes the "big picture" of how to perform a task that comprises multiple panels. Or a user may need an explanation of a difficult technical term or concept encountered while performing a task. Whereas the contextual help is area-specific, this supplementary help might be thought of as task-specific or term-specific information.

Once we step beyond the help keyed to specific panels, determining what the user wants becomes less certain. We can make a best guess and provide task information based on the part of the interface currently being used. Sometimes the guess will be right, and other times it will be wrong. When wrong, the help facility has failed the user. What we need is a way of presenting the supplementary help that is direct and easily recognizable, yet flexible enough to accommodate the varying information needs of all users and the different ways in which users might express their needs.

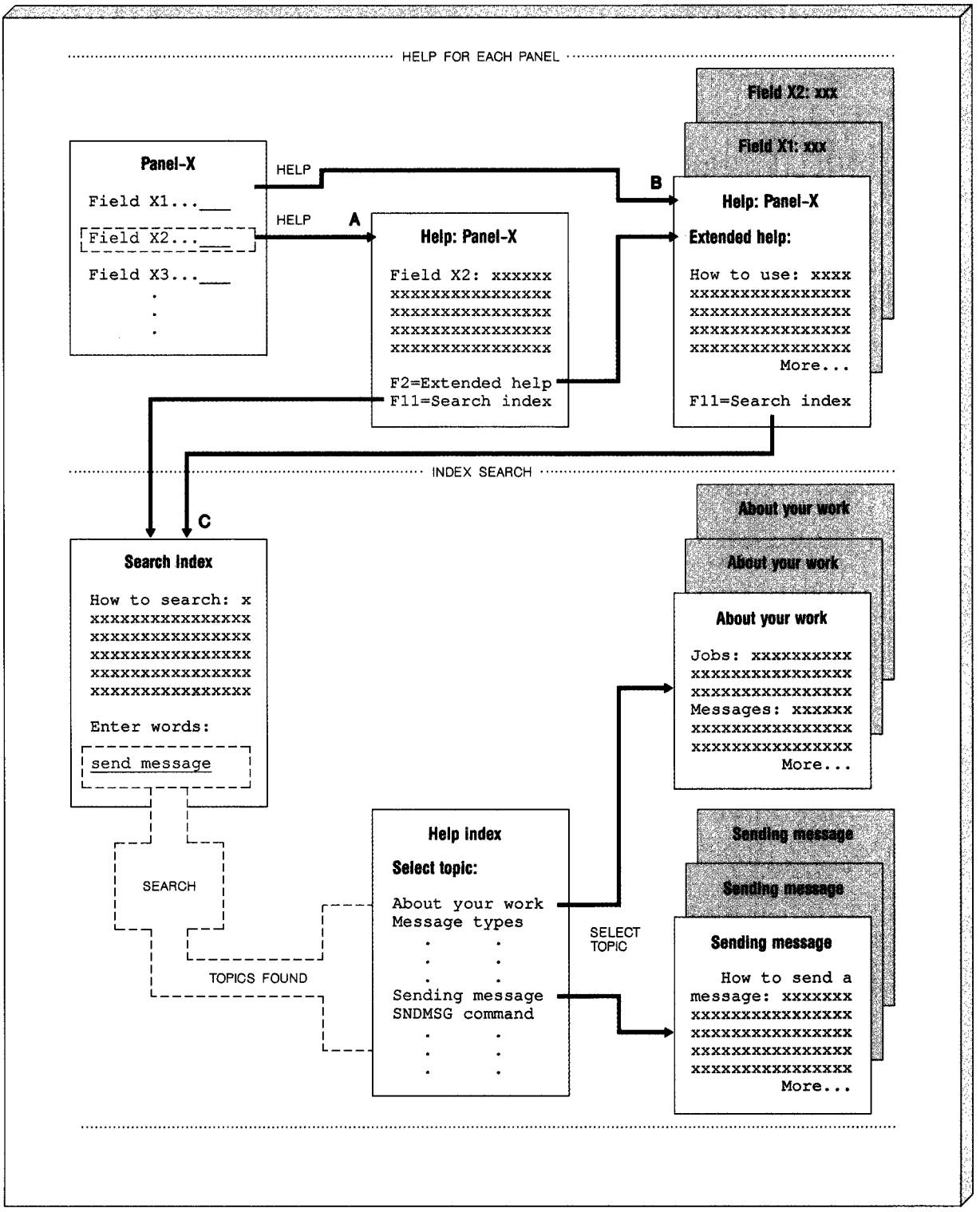
We discarded the sequential book model when we designed the help facility around the concept of independently accessible information modules. However, to provide an easily recognizable structure for the supplementary help, we went back to the book model and borrowed the concept of an index. Thus, rather than try to guess what users want, we would provide an alphabetical list of all available information from which to select one or more topics for viewing.

If this help index is small, users should have little difficulty quickly finding the needed information. If an index is large, however—as was expected for the operating system and large applications—its bulk would make it difficult to find the necessary information quickly. Furthermore, topics containing related information would likely be scattered throughout the index. We had to give users a mechanism for subsetting the index to the topics related to their current information needs only. This mechanism became known as *index search.* We would give users the capability to search the help index. Recognizing that the users might not know the proper system terminology, we would also let them do the search in their own words.

Index search, as represented in the lower half of Figure 3, can be accessed from any contextual help panel via a search index function key (path C). When a user requests index search, a panel containing explicit instructions on what to do is shown first. The user can press Enter to see the entire index or enter search words. If search words are entered, those topics in the index that match the user-supplied words are selected and this subset of the index is displayed. When the user selects a topic from the index, the text of the topic is displayed.

Although not shown in Figure 3, all help panels are also supported by help information. If the help panel contains entry fields or selections, such as the help index, pressing the help key provides field help and extended help like any other working panel. Pressing the help key from these secondary help panels then provides the general description of how to use help.

Figure 3 Help structure from the user's viewpoint



Design considerations

The concept of cursor-sensitive help combined with index search was proposed before development began on the AS/400 system. Field help had been used successfully on predecessors of the AS/400, in particular the System/36, so no validation was considered necessary for the panel help part of the help facility. However, we thought we ought to verify the usefulness of index search before committing it to development. Therefore, a prototype of index search was developed and tested, as is described later in this paper.

The help facility was designed as an integral part of the AS/400 user interface, and the user interface, including the help facility, was developed in coordination with the other system facilities. Methods of presenting the help information were constrained by the requirement that help must be accessible from nonprogrammable terminals (such as the 3270 and 5250 series of display stations) capable of displaying only fixed-size characters. This constraint limited the type of information to text only (except for illustrations that could be devised using only typical keyboard characters and punctuation marks).

Like the persons who use the help facility, those who write the information for the help facility may not be experienced programmers. Therefore, the techniques for defining the information and linking the information to panels and the index must be usable by persons with little or no programming knowledge. Many of the techniques used are similar to those used for a text markup language, such as IBM BookMaster[3] or the Systems Application Architecture (SAA) dialog tag language.[4]

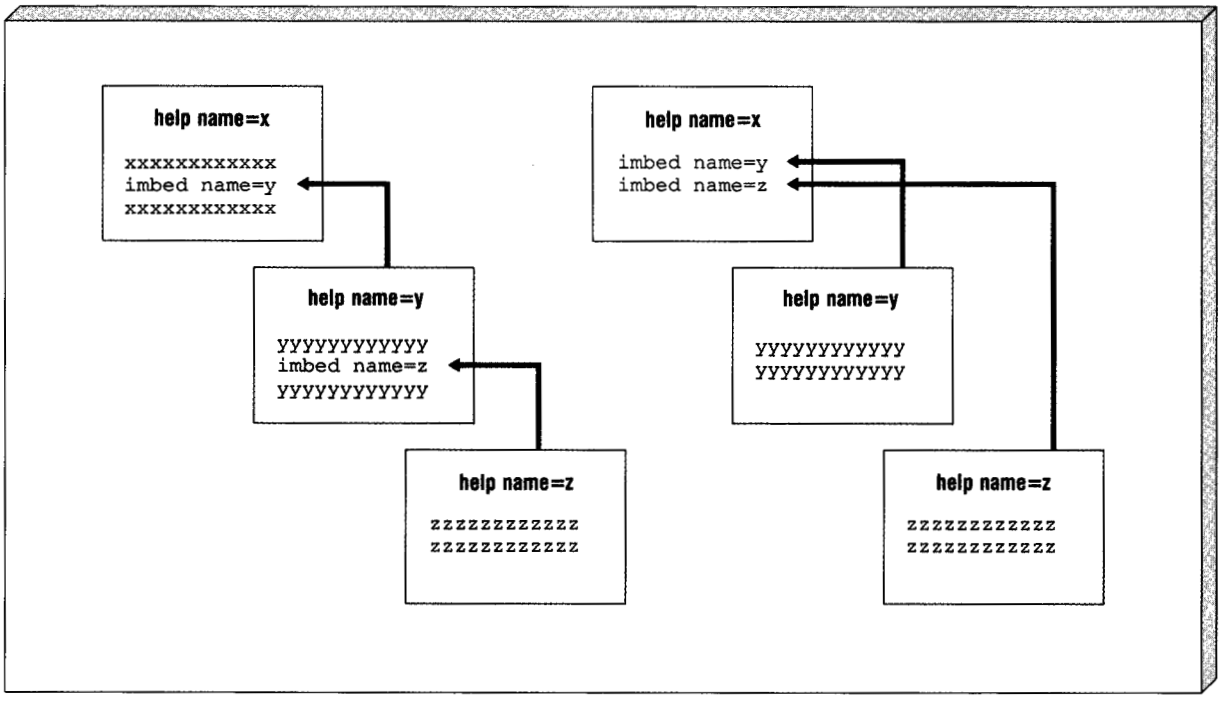
Information modules as building blocks. Each information module is a separately identifiable entity that has its own name by which it is accessed. The name is specified on the first line of the module definition. The actual help text follows on subsequent lines. A delimiter indicates the end of the module. The information modules are made accessible on the system through a simple process similar to that used to compile files and programs. The definition of each information module, which includes its name and text, is entered into a source file. To keep the source files to a manageable size, logical sets of information modules may be grouped into several source files. For example, one source file may contain operations topics, whereas another source file might contain glossary-like definitions. A *create* command is then issued for each source file, which preformats the text and defines the set of information modules to the system as a separately named system object. (See Reference 5 and Figure 12, later, for more about the object concept.) Each information module can then be displayed by specifying its name to the help facility (and the name of the object containing it if it is being accessed from a different system object).

A simple mechanism must be provided that allows the writer to link one information module to one or more other information modules.

If the information modules are to be combined in true building-block fashion, a simple mechanism must be provided that allows the writer to link one information module to one or more other information modules. The mechanism chosen was the familiar text processing technique of an imbed. One or more information modules can be inserted in another module at run time by specifying their names in an imbed statement. Multiple levels of imbeds are possible. One information module may imbed a second information module, which itself imbeds a third, as illustrated in the left side of Figure 4. Or one information module may be only a skeleton that contains imbeds of one or more other information modules, as shown in the right side of Figure 4. As is discussed later in this paper, the skeleton technique was used effectively to separate the help index from the help information accessed through the index.

Defining and displaying contextual panel help. To ensure that help support is consistent on the AS/400 system, field help areas and the type of information that should be provided for these areas are defined for each type of panel—menu, entry panel, list panel, and information panel.[5] These definitions are part of the AS/400 user interface standards. An example is shown in Figure 5. The type of information and sequence in which the information should be presented in extended help are also specified.

Figure 4 Putting information modules together through imbeds



In some cases, the type of information to be provided depends not only on the help area but also on the current condition of that area. If, for instance, a user types a command on a command entry line and requests help, help for the command should be provided. If the command line is blank, extended help that describes how to use the command line should be provided.

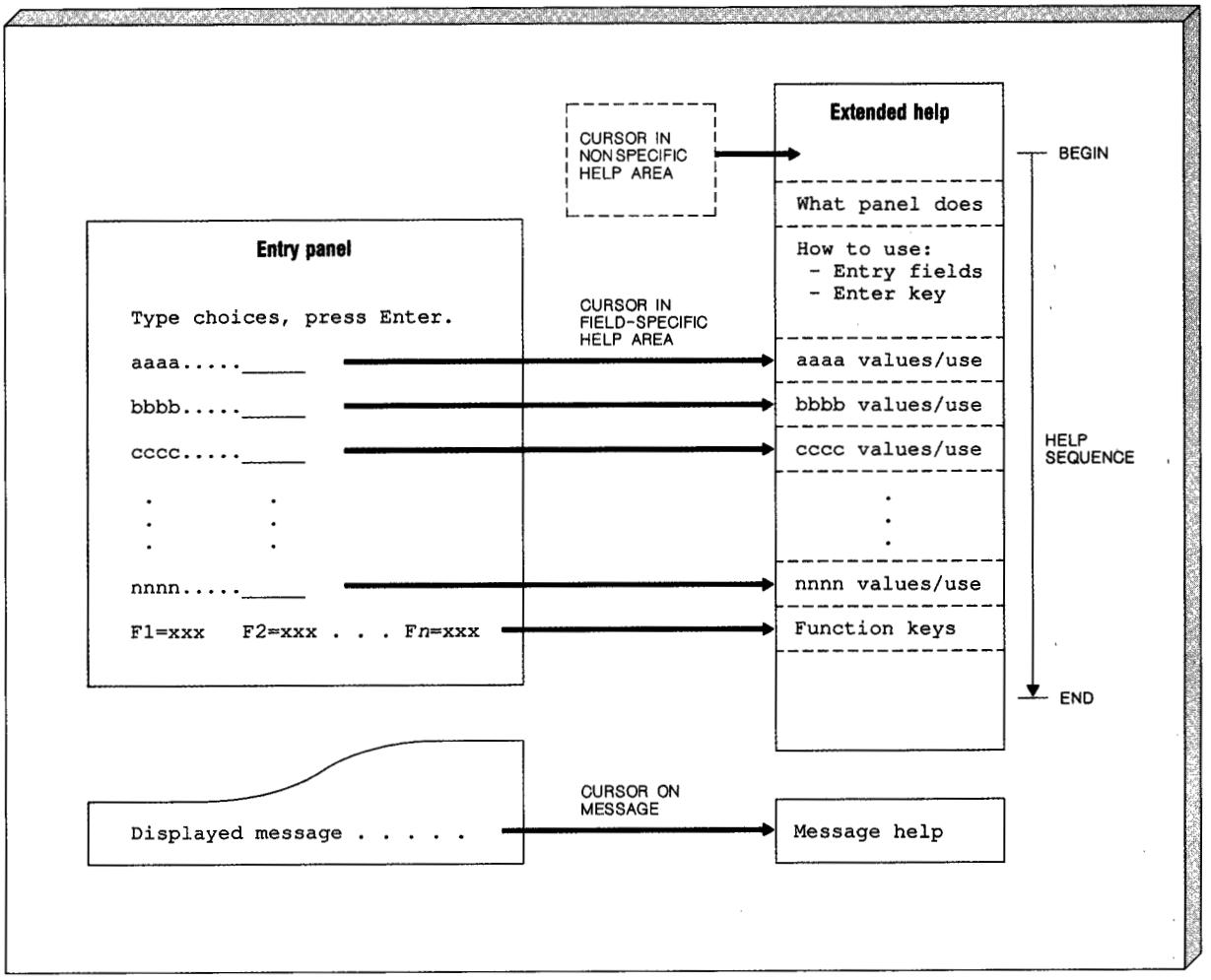
Each panel definition contains a list of information module names for the panel. The name of the information module to be displayed first is flagged, based on cursor position. When a user presses a help key, the information in the flagged module is displayed. If the cursor is in a panel area where no field help is defined, or if the user requests extended help while viewing field help, the information in all the modules listed is displayed in the order listed (as in Figure 5, for example). To the user, the information in all modules appears as a single, scrollable set of information.

Making an index search work. Information accessed through index search has the same modular structure and is defined the same way as contextual help. Instead of being linked to particular areas of panels, the information modules used for index search are linked to particular titles in the index. Because the contextual help is tied to specific areas of panels, determining what information to provide and how to provide it is relatively straightforward. With index search, such questions as the following had to be answered:

- Is an index search more effective than other commonly used techniques?
- What search process should be used to give the best chance of a user finding the topic sought, without frequent failures and while providing adequate performance?
- What method should be used for defining the search words that is both flexible and usable by the writers of the information?
- How should the information itself be designed to support and take advantage of the modular help concept?

Index search prototype. Although a searchable index appeared to be a simple solution to providing a flexible method of accessing help information and

Figure 5 Example of help area definition



search techniques had been used successfully for information retrieval, we wanted to make sure index search would be effective for AS/400 system users. Because a large proportion of the expected users would have little or no information-processing knowledge or experience, we developed a prototype to serve both as a demonstration and test vehicle. The index search prototype was tested against two alternative access methods:

- An on-line index that could be scrolled but not searched. As an aid to scrolling the index, users could position the index to any letter of the alphabet by entering that letter.
- Menus that organized the topics in a hierarchical set of tasks. Users would start by selecting a task category on the initial menu and then successively select options on lower-level menus until reaching the menu where a specific topic could be selected.

The complete set of on-line help from an existing word processing application—the equivalent of about 500 printed pages—was used for the test. Although the information was not modular, it was organized in sections and subsections that could be individually accessed.

Thirty IBM personnel volunteered as test participants, all of whom had had experience with word processing, but were not familiar with the word processing application being used for the test. Ten participants were randomly assigned to each of the three help access methods.

Each participant was asked to do 20 tasks that required the participants to find information related to a particular word processing function. As an example, one task asked participants to find out how to double space an entire document. All tasks asked for information that existed in a unique location in the on-line help. The testing verified that the searchable index approach resulted in a significantly higher success rate and a significantly faster task completion time than either of the alternative methods of accessing the information. Some of the prototype design was used in the implemented index search facility, particularly in the search algorithm.

Providing too much information could so overwhelm users that it hinders their ability to recognize the specific information they are seeking.

Search process. The goals for the search process were, to some extent, contradictory. That is, the search should be specific enough to produce only topics closely related to what the user requested, and yet be flexible enough to allow users to do the search in their own words. A search based on matching word strings would frequently fail, even if we ignored certain parts of speech, such as articles (e.g., an, the) and prepositions (e.g., of, to). The word strings users entered were unlikely to match specific word strings in the help. In addition, a string-matching search of a very large amount of information could be unacceptably slow. A keyword-matching search offered more promise, provided that an extensive set of synonyms covering the common variations of the words was also supplied.

A keyword search could produce useful results, even if not all of the user-supplied words were matched. For this reason, a pure AND search that required matching all of the significant words supplied by the user was discarded as being too inflexible. An OR search, on the other hand, would be successful most of the time, but would frequently result in such a long list that users would often have difficulty finding what they really wanted. Experience indicated that providing too much information could so overwhelm users that it hindered their ability to recognize the specific information they were seeking.

We could, of course, give users the option of doing either an AND or an OR search. If we provided both options, however, we were not only imposing an extra decision-making step on the user but also forcing some users to make a decision about two options they might not even understand. Thus we would have been failing the objective of making the use of the help facility self-evident. Here was one instance where we were willing to sacrifice function for simplicity. To search for help, users should have to decide nothing more than what they want. What we needed was a single search process that provided a compromise between an AND search and an OR search.

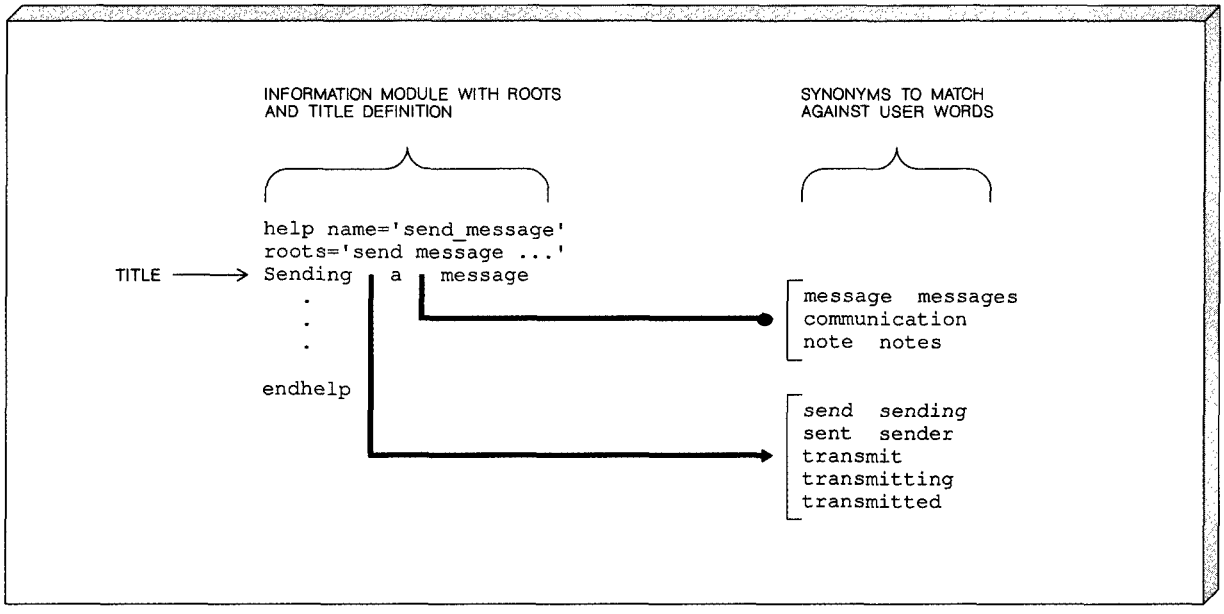
The solution, demonstrated on the prototype, was to do an OR search and then apply a best-match algorithm to the results, using the following steps:

1. Find any topic that matches any of the words entered by the user.
2. Count the number of times each topic matches the user-supplied words. For example, if a topic matches only one of the user-supplied words, give it a rating of 1, if a topic matches two of the user-supplied words, give it a rating of 2, and so on.
3. Display the titles of the topics that have the highest ratings.

Thus, if no topics are found that match all of the words, the topics that come closest to matching the user entries are shown.

Mechanism for defining the search words. A keyword-based search is only as good as the set of predefined keywords provided for each topic. The burden of success or failure is placed on the person who defines the keywords. If the mechanism for defining the keywords can be applied by persons who may not be programmers but who are experts in the art of using (and translating) language, these persons can incorporate their own intelligence into the search process. The mechanism provided is based on associating user-enterable words (synonyms) with each topic through one or more link words, called *root words.*

Figure 6 Example of root words and synonyms for one index topic



Each information module that is accessed through the help index contains a definition of the root words associated with it and the topic title displayed in the index. For each root, an associated set of synonyms is defined. The synonyms are matched against the words entered by users. The roots serve as a link between the synonyms and the topic itself, as shown in Figure 6. If a user enters a word that matches any of the synonyms defined for a particular root, any topic containing that root is flagged as matching the user-entered word. This root-synonym approach allows common sets of synonyms to be used like building blocks in any topic for which they are appropriate. For example, the root word *message* with its associated set of synonyms can be used in any topic that discusses messages. The format of the root and synonym definitions is similar to that of the ITOP and ISYN tags in the SAA dialog tag language.[4]

The help index is built through a process separate from that used to create the files containing the help text. By taking advantage of this separation, combined with the ability to imbed one information module in another, it is possible to build and modify the index independently of the help text that is accessed through the index. (See Figure 4.) This is done by defining the source for the index as a sequence of skeleton information modules that contain the definition of the root words and the topic title, as indicated in Figure 7, but no actual help text. Instead, these skeleton modules contain only an imbed statement that points to the information module containing the help text. The module of help text is accessed only when a user selects the topic from the index, not when the index is built or searched.

The skeleton information module with imbed offers opportunities for accessing other information through the help index. For example, the AS/400 control language (CL) provides an extensive set of commands, each supported by detailed on-line help.[5] Users can access this command help directly from a command entry line or prompt (form fill-in) panel or by searching the index. Command topics in the index contain imbeds that access the same information users would see if they were to request help on the commands. When a user searches the index for information about a task or term, some of the topics listed may be descriptions of related commands. This is a relatively innocuous way of telling users about commands as a fast path.

When the index is built, the order of the information modules containing the root definitions and titles in the source file determines the order in which titles are displayed in the index. If the source file is orga-

Figure 7 Format of source for building index

```
help name='aaaa'
roots='rootal rootа2 rootа3 ... rootan'
Title for topic aaaa
imbed name='aaaa_text'
endhelp

help name='bbbb'
roots='rootb1 rootb2 rootb3 ... rootbn'
Title for topic bbbb
imbed name='bbbb_text'
endhelp

help name='cccc'
roots='rootc1 rootc2 rootc3 ... rootcn'
Title for topic cccc
imbed name='cccc_text'
endhelp
    .
    .
    .
```

nized so that the titles are in alphabetical order, the index is in alphabetical order. By using the imbed technique indicated in Figure 7 so that the source file contains only skeleton information modules with root and title definitions, the sequence of the index can be easily grasped just by scanning the source file. The source file can be sorted manually by rearranging the order of the information modules in the file, or a simple program can be devised to sort the file as appropriate for each language into which the index is translated.

Application-specific indexes. A single index could be built that covers the operating system (OS/400™) as well as AS/400 applications. If an application is not installed in a particular system, however, its help information could not be accessed. Furthermore, a help index covering the operating system and all applications would likely be unwieldy, even with the search capability. For these reasons, the operating system and separate applications, such as AS/400 Office, each have their own help index. Each index contains how-to-do-it task descriptions and what-it-means explanations of terms and concepts. In addition, the operating system index contains a topic for each CL command. These command topics access the command help via imbed statements as described earlier.

When a user requests index search, the application being used when help was requested determines which index is searched. For example, if OS/400 was being used, the operating system index is searched, and if AS/400 Office was being used, the office index is searched.

Recognizing that users might want to access operating system information while using help in another application, or without going into help at all, the following alternative paths are provided to invoking index search for the operating system index:

- Start Index Search (STRIDXSCH) command
- An option on a specialized User Support menu. In addition to the operating system index, this menu provides access to various other user support functions available on the AS/400 system, including on-line education, a question-and-answer facility, and a list of AS/400 publications. This menu can be requested through a function key (F13) on any operating system menu or help panel, or on help panels in many applications.

Search example. Figure 8 shows an example of a user searching the operating system index. Note that the initial search panel contains explicit instructions on how to use index search. The user enters "send message" and receives a subset of the help index that lists the topics found. The same entry line is provided below the index, and search words entered by the user are shown for reference. To initiate another search, the user needs to change only the search words and press Enter again.

To select one or more topics from the index for viewing, the user enters a 5 in the entry field beside the topic title. A printing option is also provided for those users who still like to have a printed copy. In fact, users can print their own specialized "mini-document" of help information by entering a 6 beside the title of each topic they want to include.

Because the operating system index contains a broad range of topics, including an extensive set of command topics, a few specialized filter words are provided to help users find certain types of topics when they search the operating system index. For example, if users want the search to concentrate on how-to-do-it topics, they can enter "how" along with their other search words. Figure 9 shows an example of using "how" as a filter. This filtering effect is achieved by adding "how" as a synonym in all how-to-do-it topics, but no others. Thus, if a user includes "how"

Figure 8 Example of user search

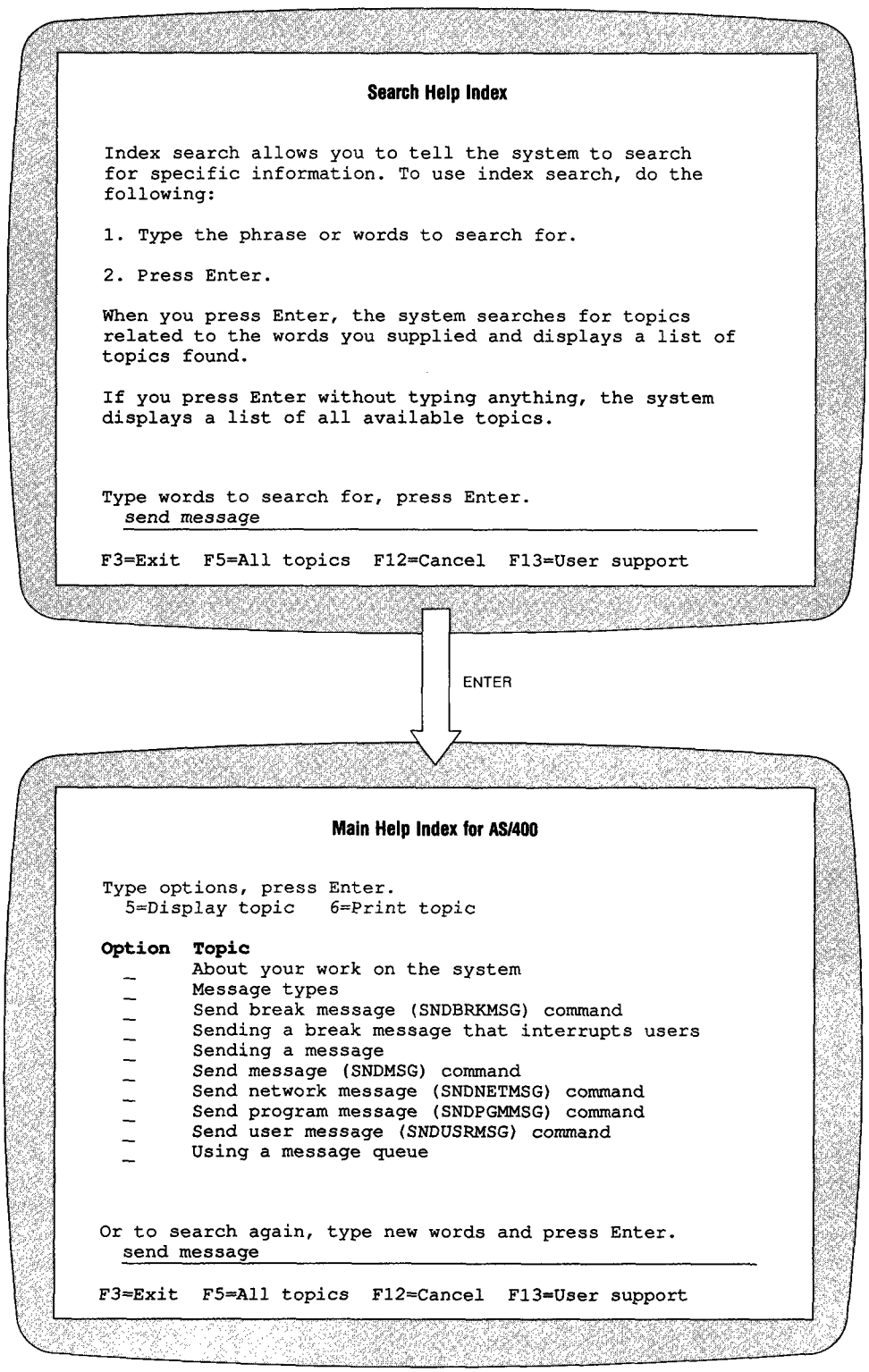
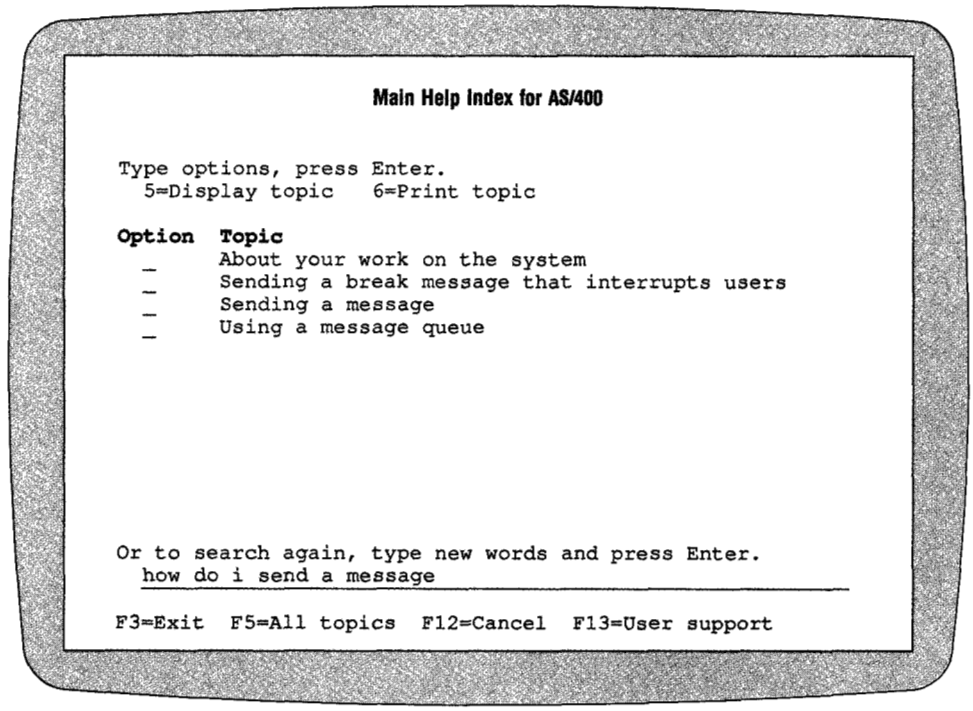


Figure 9 Restricting search to how-to-do-it topics by using "how" as a filter word



as one of the search words, how-to-do-it topics that also match the other search words have one more match than other types of topics that match the other search words. Similarly, "what" is included as a filter word in the synonyms for all what-it-means topics, and "command" is included as a filter word for all command topics. Figure 9 also demonstrates that users can enter search requests as complete statements, rather than merely keywords.

Care was taken to ensure that the most commonly asked user questions result in a useful response. Discussions with IBM systems engineers and marketing representatives indicated that questions related to finding printed output are very common. Figure 10 shows the results of searching the operating system index using the words "where is my report." Also shown on the figure are the search words that matched each topic.

Providing useful information. Providing tools for writers to create help information and users to access it will not make a useful help facility unless the information itself is useful. To ensure that the information would be useful, detailed guidelines define (with supporting examples) the type of information to provide, length restrictions, and the specific format of the source files containing the information. These guidelines are particularly important for the information to be accessed through the help indexes, because there are no obvious indicators (such as a specific area on a panel) of the type of information to provide. Following are examples of information provided in the operating system index.

Stating that the help provided through the help indexes should be either what-it-means or how-to-do-it information is an insufficient guideline. When a user selects a what-it-means topic, the topic should answer not only "what does it mean" but also "what does it mean to me" in terms the user can understand. In some cases, a simple glossary-like definition may be sufficient, as in Figure 11. Note that closely related definitions are put together, so the user sees them both at the same time. In other cases, where evaluations and interviews indicated that users are having difficulty with a concept, an expanded explanation is provided, as in Figure 12. This expanded explanation is not intended to explain the concept in detail, but to provide enough understanding to

Figure 10 Search results for a commonly asked user question

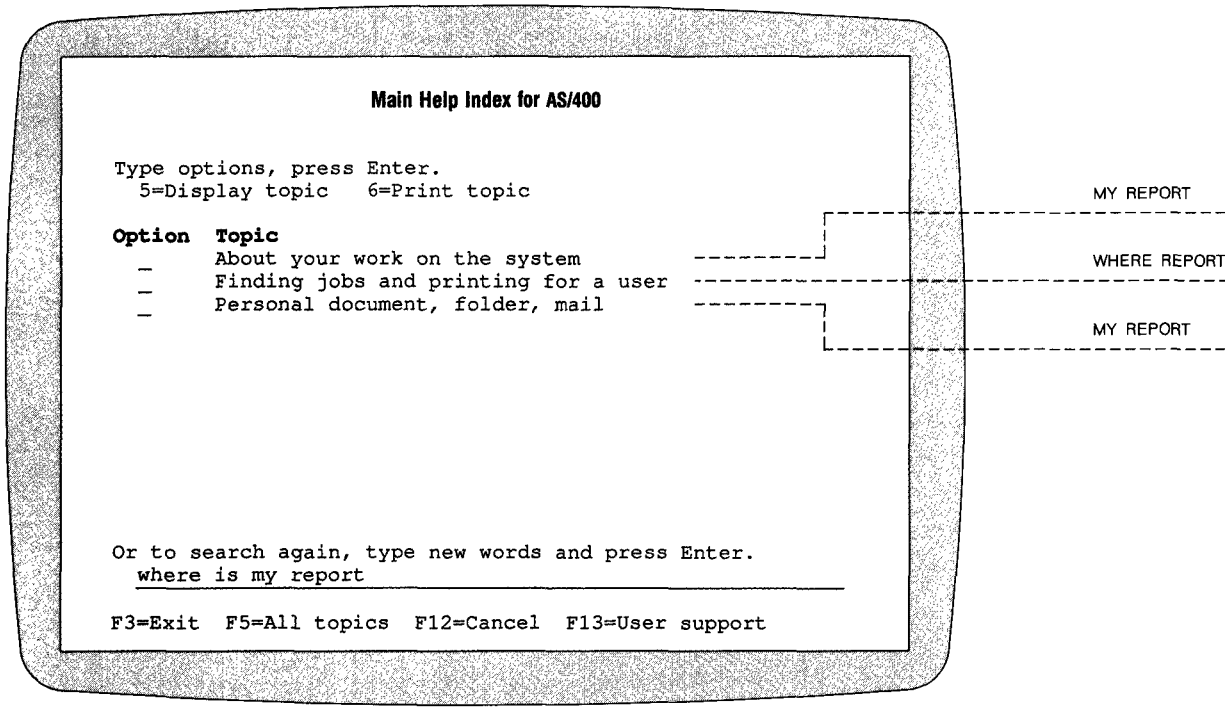


Figure 11 Index topic with glossary-like definitions

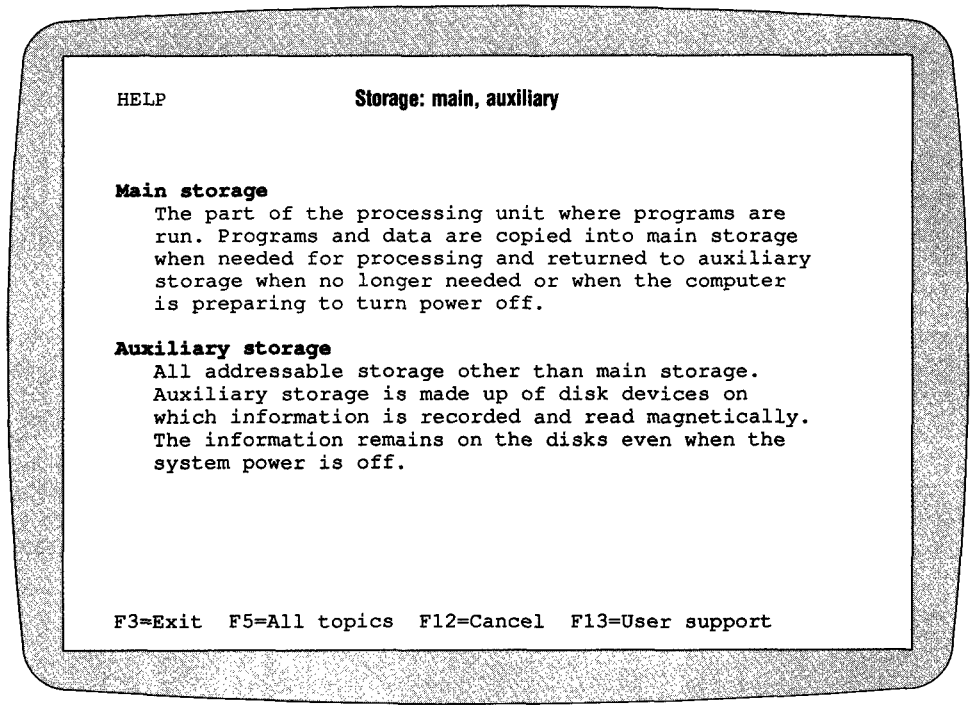
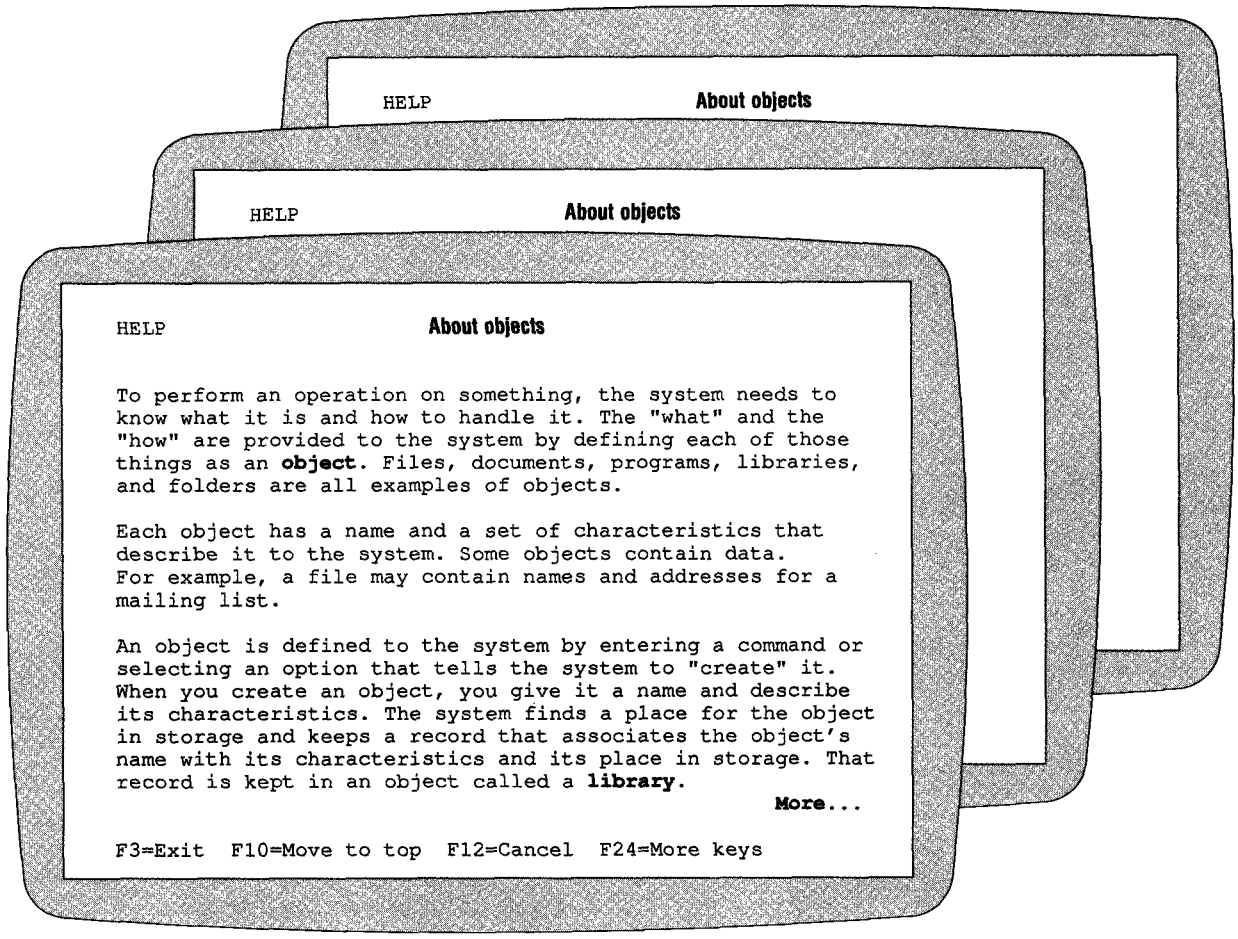


Figure 12 Index topic with expanded concept explanation



allow the user to get on with the immediate task. More detail is available through the on-line education or printed manuals.

If users ask for information about a system or application function, they usually want to know how to get it or what to do with it. Many what-is-it topics also have how-to-do-it information, as in Figure 13. Most of the how-to-do-it topics provide cookbook-like descriptions of how to do specific tasks, as in Figure 14. In addition, a higher-level view is provided by a set of task overview topics. These topics briefly describe what is involved in handling a category of tasks, such as controlling printing or using a message queue (as shown in Figure 15), and they tell the user how to get to the place (such as a menu or "work with . . ." list) from which those tasks can be done.

For all of the what-it-means and how-to-do-it topics, a minimalist philosophy is followed.[6] Brevity is stressed, both in the total length of each topic and in the way the text of the topic is presented. The what-it-means topics are generally a panel or two in length. The how-to-do-it topics are often a little longer, but even a broad level topic such as the overview of controlling printing is limited to a length of four panels, even at the expense of eliminating some information. Also stressed is the use of writing techniques that make it easier for users to find information by scanning. These include using short paragraphs, breaking up information into lists, and eliminating unnecessary transition sentences or phrases.

Evaluation and tuning. Printed drafts were used for verifying the accuracy of the help information, but

on-line evaluation was needed to verify the appropriateness of the information and the index search results. Early versions of the help information were made available on development systems. Development people as well as visiting systems engineers were encouraged to try it. Because it was new, index search received special attention. For instance, the operating system index, which contains how-to-do-it and what-it-means information on operating the AS/400, was used in informal usability evaluations.

Participants in these evaluations included visiting IBM systems engineers and customer personnel who were familiar with the operation or use of the System/36 or System/38. The participants were asked to perform a specific set of typical operator and user tasks, such as copying a file or finding an output file and moving the output to a different printer. They were asked to use index search, but the tasks they were asked to perform did not require that they use the help information.

These evaluations verified that the index search topics contained the appropriate information and helped the users complete typical tasks. Because the information design was based on the index search guidelines, the results also served to verify the guidelines. Specific changes were made to the topics as a result of the evaluations. For example, a topic on running a batch job was modified to emphasize that a program must be called to begin the job. The title of the topic "Changing the order of printing" was reworded to "Changing an output file so it prints first."

When hundreds of index search topics—many with one or more identical synonyms—are searched, the results are sometimes not as expected. On occasion, topics that are expected to appear do not show up, and inappropriate topics may show up when they should not. The solution is to tune the index. That is, selectively adjust the synonyms for specific topics and try the search again. Several iterations of trial searches and adjusting the synonyms may be required.

During the tuning process, continual compromises must be made between finding all appropriate topics and eliminating inappropriate topics. Potential impacts on translation must also be considered. An index that is perfectly tuned in English may be difficult to translate. The search example in Figure

Figure 13 Explanation topic with how-to-do-it information

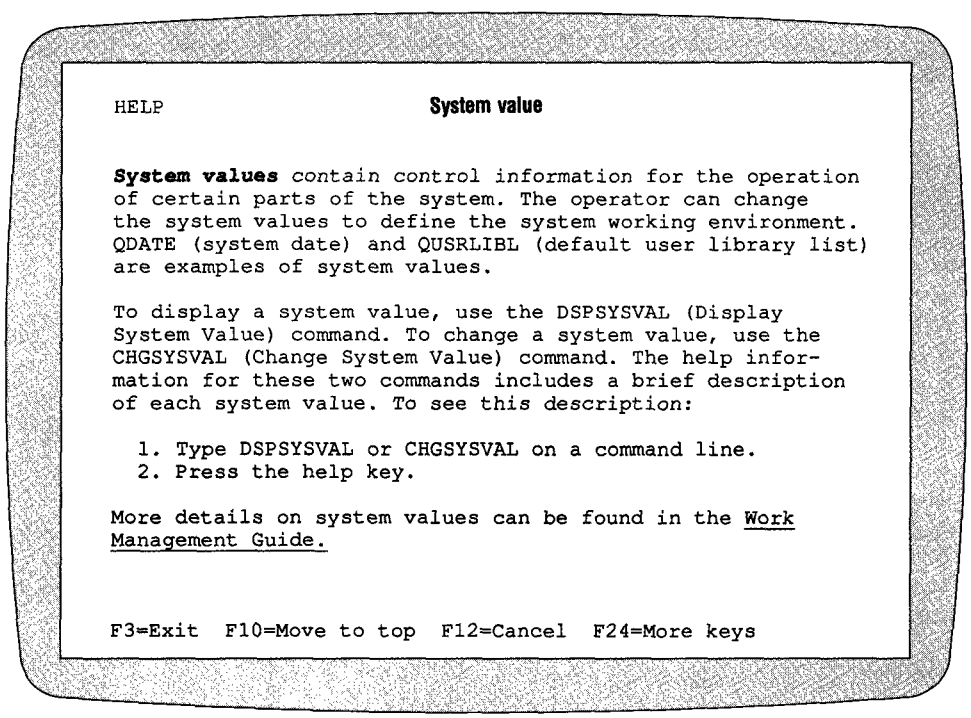
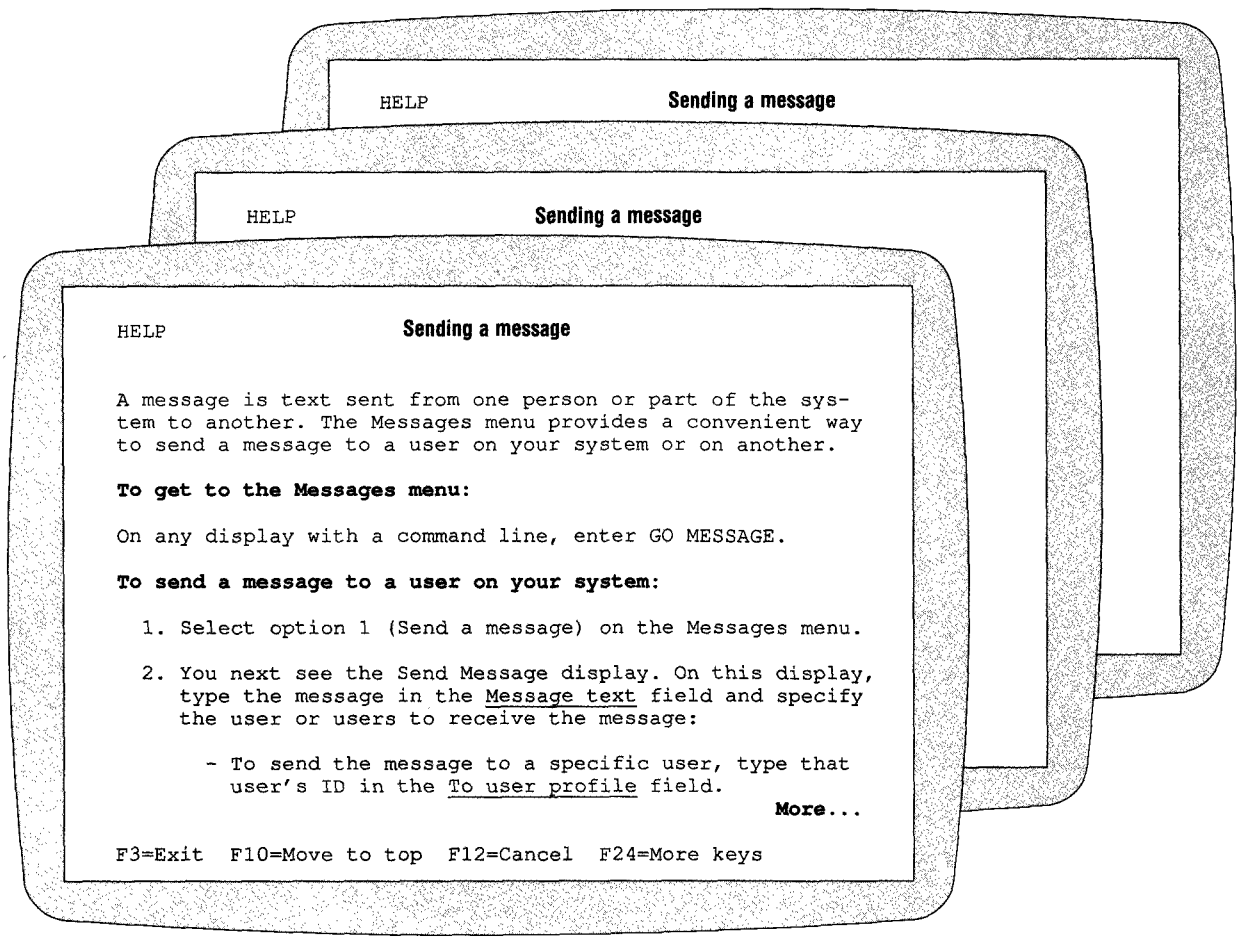


Figure 14 How-to-do-it topic for specific task



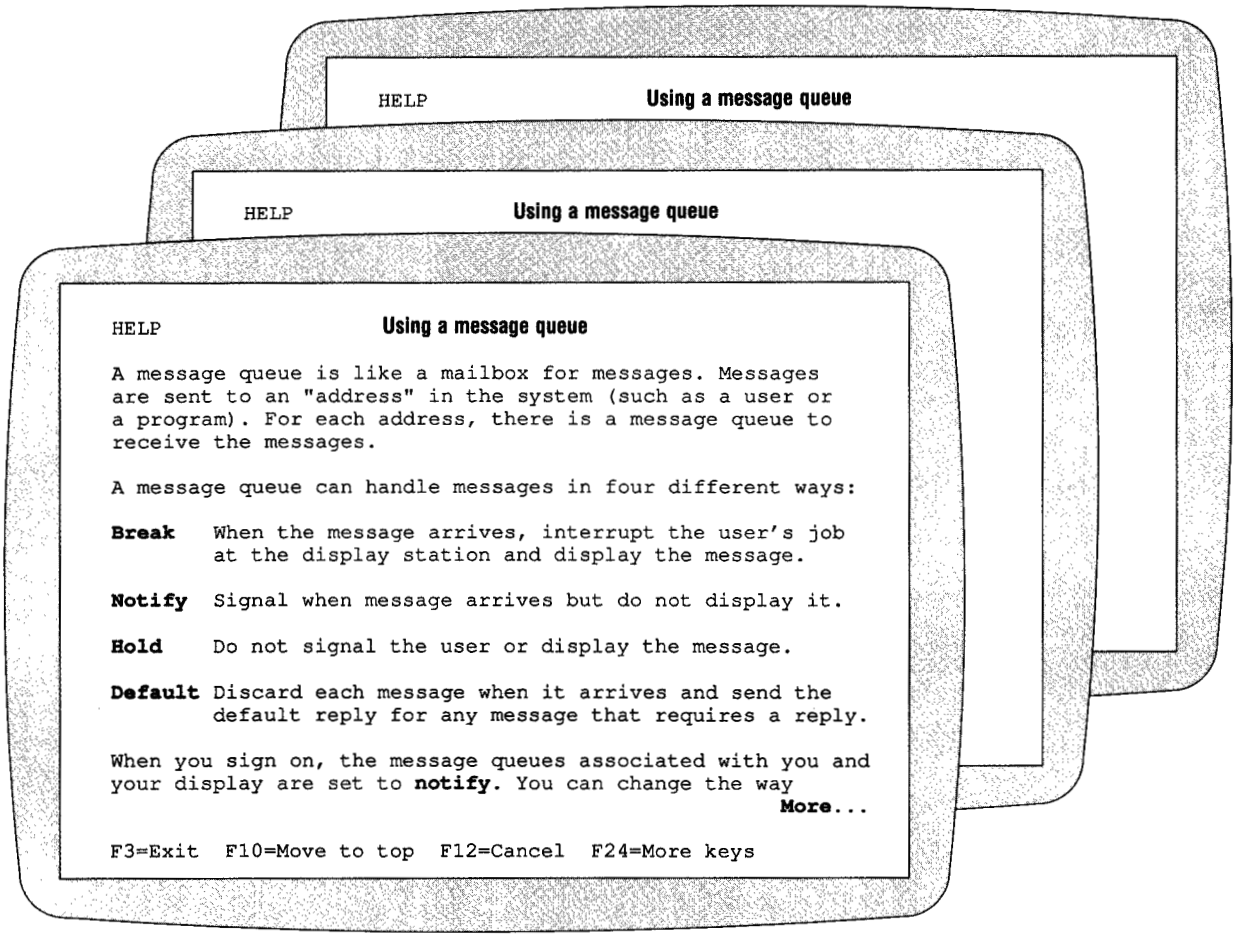
10 illustrates the tradeoffs in tuning. The first two topics are appropriate for the search words "where is my report." The topic "About your work on the system" describes how a key menu—the User Tasks menu—can be used by individual users to access and control their system jobs and printing. The topic "Finding jobs and printing for a user" describes how to determine the status of jobs and output files. The last topic, "Personal document, folder, mail," is merely a series of definitions and is not appropriate to the search words entered. Yet, to "tune out" this topic might introduce subtleties that would prevent its appearing in other searches where it is appropriate and would likely increase the translation difficulties.

Discussion

Comparison with Common User Access. Although the AS/400 help facility was conceived prior to the time the SAA Common User Access (CUA) architecture was finalized,[7] its structure closely matches the CUA help structure. As required by CUA, contextual help (both field help and extended help) and a help index are provided by the AS/400 help facility, but the help index is taken a significant step further by giving users a word-oriented search capability.

User reactions. Evaluations and responses from users indicate that the help facility is successful in helping access the information needed to complete the task at hand. Users particularly like the ability to search for information using their own words. Some observers have described the index search as being "intelligent" or "natural language." It is not, not in the commonly understood sense of employing logical inference statements to interpret and respond to user input. It does, however, give experienced writers simple methods of tailoring the search results, based on their native intelligence and language skills.

Figure 15 Task overview topic



Conclusions

The AS/400 help facility uses proven techniques of information chunking, area-specific help, and information search. Although none of the techniques can be considered novel by itself, the combination of the techniques, tempered by an overriding emphasis on simplicity, provides a flexible on-line information facility that gives users contextual help for the panel they are currently using and allows them to ask for supplementary how-to-do-it and what-it-means information in their own words. The information itself is provided in the form of small units, each of which is designed to answer the user's immediate need.

Acknowledgments

The AS/400 help facility design and development involved significant contributions of people from the IBM Rochester Programming Center, including human factors, information programming development, and software strategy personnel. The author wishes to acknowledge the efforts of Wayne Brooks and Devon Snyder, whose programming design and implementation made possible a truly modular on-line information facility with a simple but efficient search capability. Thanks are also due to Joe DiCecco and Chris Young for their design and testing of an index search prototype that demonstrated the value of user-initiated information search and provided the best-match search concept. The usability evaluations that proved so valuable to the development and verification of the index search information were the result of the work of Chuck Haydon.

Cited references

1. G. Kearsley, *Online Help Systems: Design and Implementation*, Ablex Publishing Corporation, Norwood, NJ (1988).
2. D. A. Charland, "Online documentation: Promise and problems," *Proceedings of the 31st International Technical Communication Conference*, Seattle, WA, WE158–WE161 (1984).
3. *IBM Publishing Systems BookMaster User's Guide*, SC34-5009, IBM Corporation; available through IBM branch offices.
4. *Systems Application Architecture Common Programming Interface Dialog Reference*, SC26-4356, IBM Corporation; available through IBM branch offices.
5. J. H. Botterill, "Design rationale of the AS/400 user interface," *IBM Systems Journal* **28,** No. 3, 443–464 (1989, this issue).
6. J. M. Carroll, P. A. Smith-Kerker, J. R. Ford, and S. A. Mazur, *The Minimal Manual*, Research Report 11637, IBM T. J. Watson Research Center, P. O. Box 218, Yorktown Heights, NY 10598 (1986).
7. *Systems Application Architecture, Common User Access: Panel Design and User Interaction*, SC26-4351, IBM Corporation; available through IBM branch offices.

General references

IBM Application System/400 Technology, SA21-9540, IBM Corporation; available through IBM branch offices.

B. Shneiderman, "Printed manuals, online help, and tutorials," *Designing the User Interface: Strategies for Effective Human-Computer Interaction*, Addison-Wesley Publishing Company, Reading, MA (1987), pp. 358–386.

R. E. Berry, "Common User Access—A consistent and usable human–computer interface for the SAA environments," *IBM Systems Journal* **27,** No. 3, 281–300 (1988).

A. M. Cohill and R. C. Williges, "Retrieval of HELP information for novice users of interactive computer systems," *Human Factors* **27,** 335–343 (1985).

C. S. Magers, "An experimental evaluation of on-line HELP for non-programmers," *Proceedings of CHI'83 Human Factors in Computing Systems*, Association for Computing Machinery, New York, NY (1983), pp. 277–281.

Dennis A. Charland *IBM Application Business Systems, Highway 52 & NW 37th Street, Rochester, Minnesota 55901.* Mr. Charland is an advisory information developer in the Software Strategy, Architecture, and Planning group, where he works on the strategy and design of user interface and help facilities. He joined IBM at Rochester in 1977 and was involved in the development of printed and on-line information for System/38 and System/36. He received a B.A. in journalism (with mathematics and physics minors) from the University of Minnesota in 1960. Prior to joining IBM, Mr. Charland worked as a technical writer/editor at UNIVAC, General Atomic, and the Aerospace Corporation.

Reprint Order No. G321-5368.