A BROWSING APPROACH TO DOCUMENTING DOCUMENTATION

Y. Leclerc, S.W. Zucker, and D. Leclerc

McGill University
Montreal, Quebec

ABSTRACT

Although every research facility contains a wealth of software, user information, and documentation, the extent and variability of this information make it difficult to retrieve. The most common medium—hard-copy—is especially difficult to use, with the resulting burden of information dissemination placed on the shoulders of a small group of expert users. This burden can be reduced with the aid of an on-line information retrieval system. Such a solution is attractive for a restricted research facility since the amount of information is limited enough for on-line storage to be reasonable, while its variability is limited enough for standard organizational principles to be applicable. The system we have designed provides a human-engineered framework within which a user can retrieve information organized around a loose hierarchy of keywords; he can either traverse the hierarchy or he can match keywords. Experience with an implementation of the system has shown that the two retrieval mechanisms are used differently. The former mechanism is used by novices for browsing through an area to determine what is available, while the latter mechanism is used by experts for quick access to specific technical information. As might be expected, a given user might be a novice in one area and an expert in another. Such typical users take advantage of both retrieval mechanisms. In short, the system provides a comfortable and efficient interface to documentation. And, perhaps more importantly, it demonstrates that such interfaces can be built and maintained with only modest expenditures in manpower and resources.

CMCCS '81 / ACCHO '81
The above problems. It is also our experience that these problems of information dissemination are not unique to our laboratory, but rather are more often the case than the exception.

2.0 SYSTEM DESIGN

The primary goal of the Browsing System was to provide a means for browsing through and/or quickly retrieving information about the laboratory's resources. It was important for our needs, however, that the implementation, and especially the maintenance, of the system require a minimum of man-power and on-line storage.

The design decisions that were made to meet these goals are described in the following sections.

2.1 Accessibility

Since executable programs are an important part of the resources of the laboratory, it was felt that, ideally, the documentation system should be accessible from within other programs, and that other programs should be accessible from within the documentation system. For example, the former might be useful when editing a program (e.g., to find the parameters to an external subroutine), and the latter would allow a person to discover what programs are available without leaving the environment of the documentation system. We have incorporated only the latter capability in the current implementation, but we see no major problems in incorporating the former in future implementations.

2.2 Information Organization

A capability for browsing requires both an appropriate presentation of individual items of information and an organizational context that will facilitate the browsing. The first need was met by naming each item of information explicitly, so that it could be retrieved immediately once its name was located. In other words, each item of information, be it an executable program or some form of text, was assigned one or more keywords. The second need was met by making each entry short enough that many could fit on one page of a normal display terminal, and these keywords were then embedded in a structure whose category names were also keywords.

We chose a multiple-parent hierarchy (i.e., a directed acyclic graph with a single root node) as the structural organization of the
keywords for its simplicity and flexibility. A strict hierarchy was considered to be too inflexible, because it forces each keyword to be placed in a specific category. This often leads to difficulties both in searching and in categorizing information. At the same time, an arbitrary network of keywords, although very flexible, is difficult for a person to grasp and follow. The compromise of a hierarchy with multiple parents allows items of information to be found through multiple paths, and yet is simple enough that traversal of the graph is natural.

The structure is implemented as a doubly linked list of variable length records containing the keyword name, the file name, and the links to the fathers and sons of the keyword.

2.3 Separation of Organization and Information

The hierarchy of keywords is a separate file, with the information pointed to by file names. This has several important advantages over storing the information within the structure itself:

1 - Once a keyword has been entered for a particular item of information, the information can be updated by the person responsible without the intervention of a "librarian".

2 - Information need not be modified in any way to be incorporated into the Browsing System.

3 - The information can be created independently of the system without the need for special editors, thereby reducing the implementation cost.

4 - Since information is not stored explicitly, the structure is relatively small and can be quickly and efficiently traversed by the system, allowing quick traversal of the graph by the user.

2.4 Human Interface

A constraint on our design of the human interface to the Browsing System was the requirement that it be usable from a standard video terminal with minimal graphic capabilities. Inspired by the standard LISP pretty-printed form of displaying lists, we chose the indented form of displaying a hierarchy illustrated in figure 1. The desire for quick traversing of the graph led to single keystroke commands, which are listed at the bottom of the screen as a reminder to the user. Figure 2 illustrates the initial display of the Browsing System. (The design of the human interface was also inspired, in part, by an application of the Zog system [3,4] to browsing through books and technical reports [5]. In fact, a version of the Browsing System could have been implemented in Zog, but this conflicted with our requirements for simplicity of implementation and efficiency of computation. See also [6], where the issues of a document retrieval language are described in detail.)

Note that only one level of the hierarchy is initially visible in Figure 2, with ellipses to indicate that further levels exist. This is to simplify the user's initial view of the system, but can be changed to view up to 4 levels of the hierarchy simultaneously. If the displayed hierarchy does not fit completely on the screen, the user can scroll the display up or down. Figure 3 illustrates a case in which 2 levels of a hierarchy are displayed simultaneously, and figure 4 shows the displayed hierarchy scrolled down several lines.

The user moves down the hierarchy by moving the pointer (in the top left-hand corner of figure 2) up or down on the screen to the desired keyword, then typing "s" to view its sons. The user can move up the hierarchy one level at a time (retracing his steps down the hierarchy), or he can move directly back to the top. The user can also locate a specific keyword by entering the full keyword, or just an abbreviation. The Browsing System then searches for the first matching keyword from the top of the hierarchy (using a depth-first search). If the matched keyword is not the one that the user wanted, he can force the system to go on to the next match.

Once the user has placed the pointer at an appropriate keyword, the associated information can be displayed on the screen, or the associated program can be run. The user can always return to the Browsing System via a control character, giving him the freedom to experiment.

2.5 Creating and Modifying a Network

Any user can create and modify his own Browsing Network, although a password is required to modify the (default) system network. A different program is used for this which is a superset of the normal Browsing System. That is, every command in the Browsing System is also available in the Browsing Editor, along with extra commands to create, insert, delete, and modify keyword entries. Figure 5 illustrates the initial display of the Browsing Editor, along with a brief summary of the available commands.

The approach used in building the system
The multiple-parent feature of the system was implemented for the two major reasons initially planned for:

1 - To browse through available documentation, be it for hardware, software, technical reports, etc., in order to discover what is available.

2 - To quickly find technical documentation about a specific piece of software (e.g. for the parameters to a subroutine).

The Browsing System has been very helpful in alleviating needless duplication of effort by providing a means for discovering, in a simple manner, what other people had done. And, perhaps more importantly, the development and maintenance costs were quite reasonable (2-3 man-months for designing and implementing the system), demonstrating the feasibility of implementing useful documentation systems at reasonable costs. When these costs are compared with the time that our expert users no longer have to expend to simply disseminate information, the investment seems emminently worthwhile.

REFERENCES


ACKNOWLEDGEMENTS

The authors are grateful to John Mohammed, David Kashtan, Harold Hubbschman, and Peter Sander for their help in designing the Browsing System, and for their critical comments and helpful suggestions concerning this paper.
Figure 1. (a) A small hierarchy. (b) The same hierarchy displayed in the indented form used in the Browsing System.

Press "i" for information. n =
↑
↓

LATEST ENTRIES . . .
AREAS OF RESEARCH . . .
SUPPORTED SOFTWARE . . .
AVAILABLE HARDWARE . . .
REPORTS & THESES . . .
DEMONSTRATION PROGRAMS . . .
PEOPLE POINTERS . . .
SUGGESTION BOX . . .

Figure 2. The initial display of the Browsing System. The ellipses following the keywords indicate that the keyword has sons. Attached to each of these keywords is an arbitrary text file, which can be retrieved by moving the pointer (currently pointing to the topmost keyword) to the appropriate keyword and then pressing "i". The set of available commands are listed at the bottom of the screen.

CMCCS '81 / ACCHO '81
**Figure 3.** Two levels of the keyword SUPPORTED SOFTWARE. The "MORE..." indicates that the two levels could not fit completely on the screen.

**Figure 4.** Two levels of the keyword SUPPORTED SOFTWARE scrolled down several lines. Notice that the arrow now points to the keyword PRINTHELP, and that the system is indicating that more of the hierarchy can be displayed by the two symbols "MORE..."
Press "i" for information.

(a) Add - Add the Input node to the network as the son of the "-->" node.
Back - Move the "-->" back one page (22 lines) when possible.
Create - Create a new node (call this the Input node).
Delete - Delete the "-->" node, and all nodes isolated by this deletion.
"downarrow"* - Move the "-->" downwards (when not available, use linefeed).
Exit - Exit from the program, saving all modifications in a new file.
Father - Go to the father (super-category) of the current node.
Generations* - Specifies depth of sub-categories seen simultaneously. Max (4).
Help - Print this message.
Information - Print the information associated with the "-->" node.
Kill link - Kill (delete) the link from the Input node to the "-->" node.
Locate - Search for the first match of the specified string in the net.
Modify - Modify the "-->" node, and make it the new Input node.
Next - Search for the next match of the string in the net.
Page - Move the "-->" forward one page (22 lines) when possible.
Quit - Exit from the program without creating a new file.
Run - Run the program associated with the "-->" node.
Sons - List the sons (sub-categories) of the "-->" node.
Top - Go to the top of the tree.
"uparrow"* - Move the "-->" upwards (when not available, use backspace).
Update - Update the network file.
? - Print the names of the files associated with the "-->" node.

(b) Figure 5. (a) The initial display of the Browsing Editor.
(b) A summary of the commands available in the Browsing Editor.

CMCCS '81 / ACCHO '81