The Effect of Advice Message Location on User Performance

S. G. McLellan, A. W. Roesler, and A. L. Elliot

***Abstract*—In 1993, a study was begun with a large commercial oil and gas software interpretation system to determine the feasibility of a general taxonomy of on-line help content and a corresponding taxonomy of human interface access methods to this content. The preliminary findings from this work were encouraging and indicate that a taxonomic approach makes it easy both for help providers to understand what information they need to supply, and for help users to find the help they need quickly. Part of this taxonomy of help content includes inside application messaging. Existing studies of on-line help messaging indicate that both user-initiated or system-initiated advice messaging can improve user efficiency by prompting users with information about what something on the interface is, what it does, or what to do once it has been activated. This study examines the placement of on-line help messages in a multiwindow software application on user performance. Subjects were automatically timed as they performed two sets of tasks: one where help messages always appeared at the bottom of an application's main window and one where help messages appeared at the bottom of the current window in focus (either the application's main window or a secondary popup window). Half of the subject group were shown one message location first; the other half were shown the second message location first. The results demonstrated that, while the order in which the subjects were provided the two messaging schemes was not significant, users performed significantly better when the messages were placed in the current window in focus, and that this performance decreased as the size of a secondary window with current focus increased. Further, a majority of users showed a strong preference for help messages located in the current window in focus.**

CAN the placement of on-line help improve user performance? This was the question we asked ourselves in 1993, when studies to determine the taxonomy of on-line help content showed that a relatively high percentage of help information requested by users could be displayed inside the application.

A review of on-line help literature reveals an expanding set of theoretical and practical answers to many of the design questions software developers are now asking about help. One striking thing about this research is how little of it deals first with what content end users of software applications actually require and, then and only then, with what access methods should be bound to what kinds of help [1].

Manuscript received May 1995; revised September 1995.

S. G. McLellan and A. W. Roesler are with Schlumberger Engineering Center, Austin, TX 78720-0015 USA.

A. L. Elliot is at Georgia Institute of Technology, Atlanta GA 30332 USA.

Publisher Item Identifier S 0361-1434(96)02285-0.

In 1993, a project was launched with a large commercial oil and gas software interpretation system to determine the feasibility of a general taxonomy of on-line help content and a corresponding taxonomy of human interface access methods to this content. The preliminary findings from this work were encouraging, and indicate that a taxonomic approach makes it easy both for help providers to understand what information they need to supply and for help users to find the help they need quickly [2].

Included in the taxonomy for on-line help were the following two categories:

- Information that is user-directed and answers the question "what is this/what does this do" about a component in the application's human interface. User-directed help means that the user determines and initiates the kind of help desired—for example, by directing a cursor or keyboard focus on a particular interface element and activating help.
- Information that is application-directed and answers the question "what do I do next" about what operations are available once a particular interface element—for example, a button—has been activated. Application-directed means that the application determines and initiates the kind of help needed.

The original study was divided into three separate experiments. Experiment 1 provided a set of questions asked by users needing help on a specific application. The experiment took the form of a recorded Wizard-of-Oz study, whereby a tutor (Wizard) received and answered all questions for help from subjects performing the same set of prescribed tasks on an application [3]. From the questions produced in Experiment 1, Experiment 2 showed that a substantial percentage of these recorded questions for help could be classified by help providers, using a proposed taxonomy, as either "what is" or "what's next" questions (41%).

This result was confirmed by Experiment 3, a repeat of Experiment 1 except for the substitution of an implemented help system for the Wizard. Here, too, 38% of the help required by subjects fell into these two categories of help.

Both "what is" and "what's next" categories of help had been implemented as messages displayed always at the bottom of the application's main X window interface. Since the percentage of help information that was requested by subjects and that could be displayed inside the application itself was substantial, and since users needed help on components of secondary dialog boxes as well as the primary window, it made

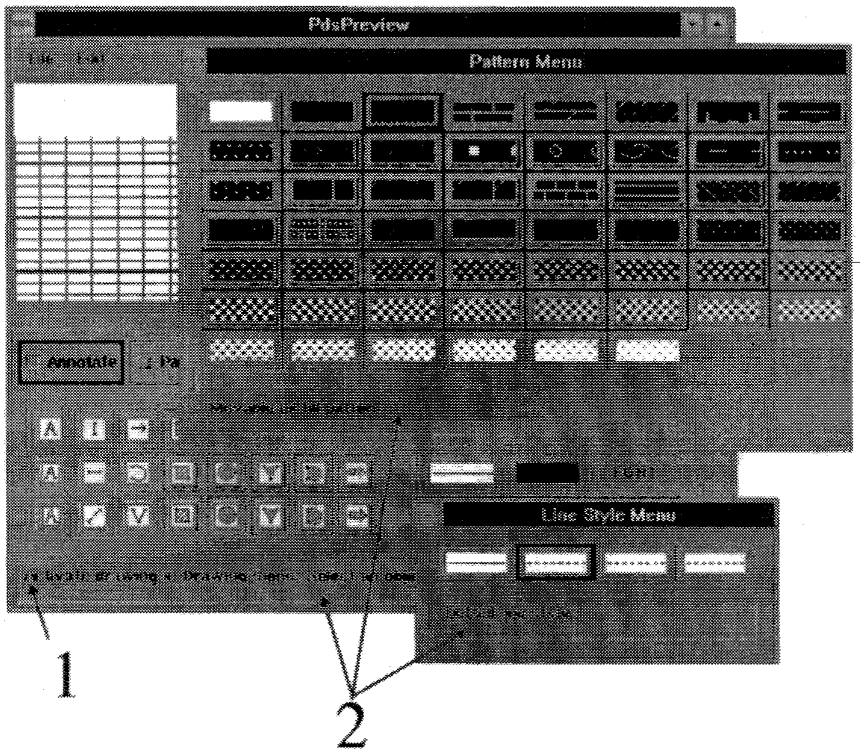

Fig. 1. Two help message implementations.

sense to analyze whether the aiding interface could itself be improved to enhance user performance [4]. Specifically, could modifying the location of help message information displayed inside an application improve user performance? The primary purpose of the present study was to answer this question by comparing the original implementation (help messages always at the bottom in the application's main window) and an alternative implementation (help messages in either the main window or secondary dialog boxes, depending of where user was).

Asking this question, however, raises several others. Would screen clutter and shift of focus significantly increase the amount of time required to make use of message help? With respect to increased time due to attention shifts, studies calculate the time required for eye movement to be approximately 50 ms and the time required to regain focus to be between 100–750 ms [5]. The overall time difference less than 2 s alone might not warrant changes to the interface and interface standard. A second question, do users have a strong preference for one alternative over the other, is, thus, important, as a strong user preference in combination with supporting performance statistics might justify changes to the interface.

METHOD

The present study focuses on the comparison of two alternative help messaging implementations. In the first implementation (1 in Fig. 1), help messages are always displayed in the message field at the bottom of the application's main window, regardless of whether the user's current cursor or keyboard focus is the main window or a secondary dialog box. Inquiries about components in a secondary dialog box would, therefore, appear at the bottom of the main window. In the second implementation, help messages are displayed in the message field at the bottom of the current window—that is, the window where the user has cursor or keyboard focus (2 in Fig. 1).

Subjects

Ten subjects were selected from among a customer community with a real use for the sample application. A pretask questionnaire gathered demographic information about the subjects and their particular computer experience (e.g., X/Motif, graphic software packages, etc.). Subjects were divided into two groups of five subjects each. To balance the possible effects of a learning curve and to counterbalance possible ordering effects, one group performed the first set of tasks with the first help messaging implementation, where messages always appeared in the application's main window. The second group performed the first set of tasks with the second help messaging implementation, where messages appeared in the application's main window or secondary dialog boxes, depending on the subject's current focus.

The Experiment

Two sets of four tasks each, or eight tasks, were presented to each subject. While the tasks reflected the type of tasks *that* would be performed during normal duties, they were constructed specifically to require interaction with the help messaging system. Before each set of tasks, the subject was instructed on where the help messages would appear. The first set of tasks is provided below.

1) In the main window, find the button that displays the message, "Selects the fill pattern for the interior of objects." Click on it to bring up the Pattern dialog box.
2) Starting from the leftmost button on the second row of the pattern palette, click on each button until you find the button that displays the message, "Bituminous or asphaltic fill pattern." Click on it to set the pattern and close the dialog box.
3) In the main window, find the button that displays the message, "Sets the style (dashing pattern) for drawing a line or border around object." Click on it to bring up the Line Style dialog box.
4) Starting from the rightmost button in the line style palette, click on the each button until you find the button that displays the message, "Dashed line style. Click on it to set the line style and close the dialog box."

Analogous to the first, the second set of tasks, given below, was used to compare performance time and user preference.

1) In the main window, find the button that displays the message, "Selects the fill pattern for the interior of objects." Click on it to bring up the Pattern dialog box.
2) Starting from the rightmost button on the third row of the pattern palette, click on each button until you find the button that displays the message, "Anhydrite fill pattern." Click on it to set the pattern and close the dialog box.
3) In the main window, find the button that displays the message, "Sets the style (dashing pattern) for drawing a line or border around object." Click on it to bring up the Line Style dialog box.
4) Starting from the leftmost button in the line style palette, click on each button until you find the button that displays the message, "Dotted line style. Click on it to set the line style and close the dialog box."

Measurements

As stated earlier, a pre-experiment questionnaire was given to each subject to show that computer expertise either was balanced across subject groups or had no significant effect on performance. Code was inserted into application software to automatically time subjects as they performed tasks on the interface, to link these activations along with a unique name for the interface component, and to record all activations by subjects during the length of the experiment. Following each experiment session, each subject was asked to complete a questionnaire to gauge personal opinions about the two help messaging alternatives and, specifically, to determine if subjects showed a significant preference for one implementation over the other. In addition, all sessions were videotaped to supplement the data measured by the timing software and collected in the questionnaires.

Of particular interest were three measurements that would be determined from the automatic timing data and questionnaire responses:

- Task performance times, calculated as the time between the first inquiry about an application interface component (detected when a subject depressed the left mouse button over a component), and the activation of the correct interface component (detected when a subject clicked *and* released the left mouse button over the component).
- User preference, indicated on the post-experiment questionnaire as a choice of which of the two help message implementation was preferred, the strength of this preference (range from "strongly preferred one over the other," "preferred one over the other," or "two systems were about equal"), and any comments about either help implementation.
- Error rates, indicated from the videotapes and timed task logs.

Related to task performance were measurements for the effects on subject performance by three key factors:

- The size of the secondary windows (dialog boxes) that, when displayed, obscured a portion of the application's main window. Here, a small dialog box of options (Line Style Menu window) and a larger dialog box (Pattern Menu) were used (see Fig. 1).
- The location of the messages, displayed either always at the bottom of the application's main window or at the bottom of the window (main or secondary dialog) that had current subject context (e.g., where the subject was working). Here, all subjects were asked to perform the same type of tasks for each of the two help messaging implementations.
- The order in which subjects were exposed to the help messaging implementation. Here, half of the subjects were exposed to one help messaging implementation first; half were exposed to the other help messaging implementation first.

RESULTS

The results are presented here in the order of the experiment.

User Group Profiles

Based on the pre-experiment questionnaires, none of the subjects had used the particular application beforehand, though all users had used the system to which the application belonged, a system where the first help messaging implementation was the standard. While the subjects represented a normal range of expertise with such an application (e.g., maximum 4 years of Motif experience versus 0.75 minimum, maximum 25 years experiences with logs versus 0 years minimum), the subjects were not grouped by computer expertise. The summary of analyses of variance revealed, however, no significant differences between groups ($p < 0.9$). Thus computer expertise was balanced across groups, or it had no significant effect on performance, or both.

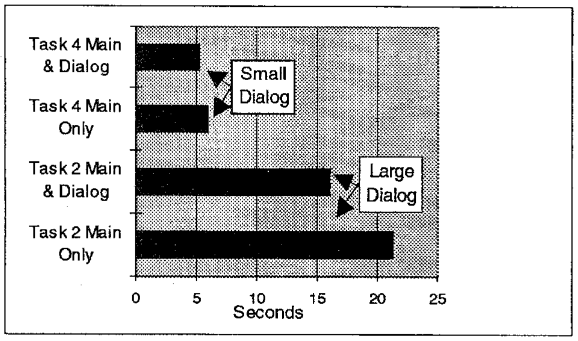

Fig. 2. Average task performance times by size of secondary window.

Task Performance Times

The analysis of the average performance data is presented in Fig. 2 along a single axis based upon the size of the dialog box. This division is based upon the hypothesis that larger dialog boxes would obscure more of the application's main window than would smaller dialog boxes. Thus it was important to determine if average performance times differed according to the size of the dialog box. For large dialog boxes, the help messaging implementation where messages were displayed in the window where the subject had current context had a 5-s average performance advantage. The small dialog box produced only a slight 0.75-s average performance advantage for this same help implementation (see Fig. 2).

In addition to average task performance times, the task performance times for each subject were analyzed. As shown in Fig. 3, 7 of 10 subjects performed the second task, selecting a particular button out of a larger dialog box, more quickly when the messages appeared at the bottom of the current dialog box. Of the 3 remaining subjects who did not perform more quickly, 2 subjects performed less than 1 s slower, and 1 performed approximately 6 s slower because of an accidental activation error. Half of the 10 subjects performed the fourth task, selecting a particular button out of a small dialog box, more quickly when the messages were displayed directly in the current dialog box (see Fig. 3). Of the remaining 5 subjects, 3 performed less than 1 s slower, and 2 performed approximately 5 s slower, again, a direct result of accidental activation errors.

An analysis of variances of measurements also showed the following:

- The effect of window type (small or large dialog box) within subjects was significant ($p < 0.001$).
- The effect of message location (whether messages were displayed in the current window or only in the application's main window) within subjects was significant ($p < 0.05$).
- The effect of ordering (exposure to one help messaging implementation before the other) within subjects was not significant ($p < 0.9$).

Learning and Error Rates

Thinking of the first set of tasks as training, we expected to see some effect of learning for each subject. Fig. 4 shows that

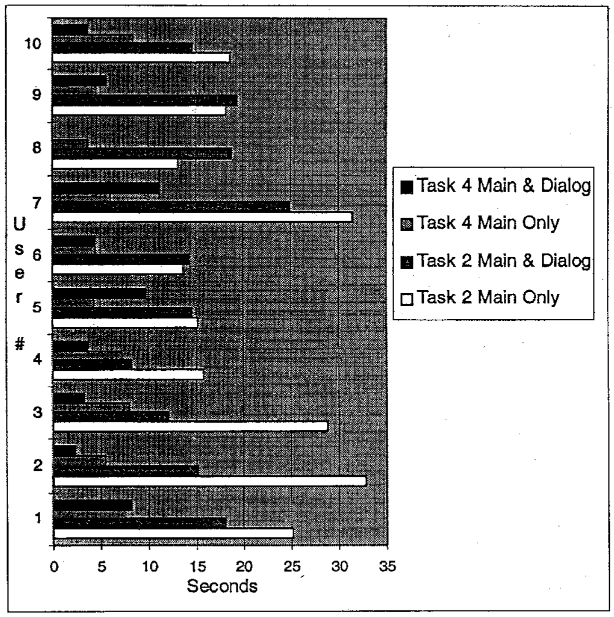

Fig. 3. Individual subject task performance times.

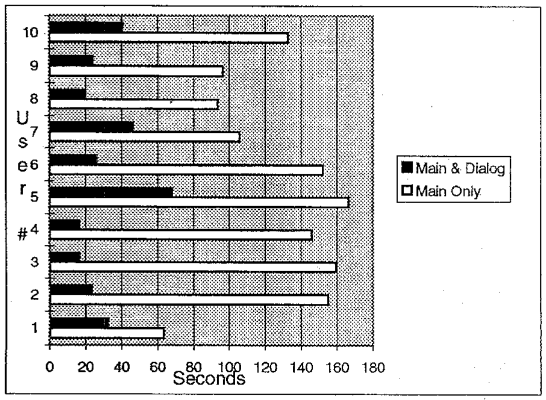

Fig. 4. Effects of learning.

subjects took longer to complete the first task set than they did to complete the second set of tasks. Users 1 through 5 used first the help messaging implementation, where messages were always displayed in the application's main window, and users 6 through 10 used the other help messaging implementation first. As shown, even while learning, subjects performed tasks faster when messages appeared first in their current context.

An analysis of errors focused accidental activation errors while browsing for help information. The total number of errors per task and the trend of error rates are shown in Fig. 5. While 8 of 10 users did commit at least one accidental error, there was no evidence suggesting any correlation between the location of the help messages and the errors committed. Short as the experiment was, however, as subjects worked with the particular help messaging implementation, they did commit fewer errors. Eight of 10 subjects committed errors only during the first four tasks, and 7 of 10 subjects committed errors only during the first two tasks.

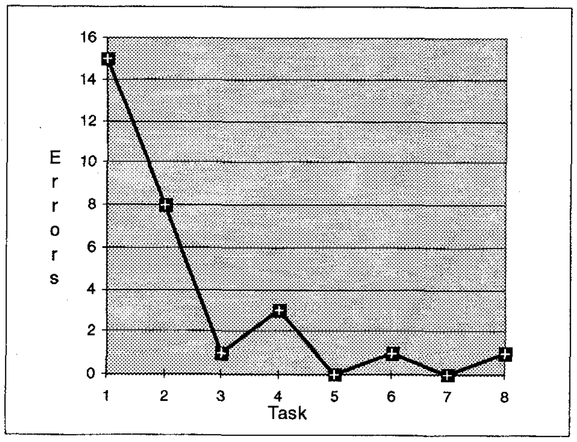

Fig. 5. Errors committed during task performance.

User Preferences

The data gathered in post-experiment questionnaires showed a clear preference for the help messaging implementation where messages were displayed in the window in which the subject was working. In fact, 9 of 10 subjects indicated this preference. Of these 9, 6 indicated a strong preference. The 1 subject who preferred messages always in the application's main window rated both help messaging implementations as about equal.

Subjects were also asked to convey any other comments they had about either or both help messaging implementations. Fig. 6 contains representative comments (bracketed information is provided for clarity). Subjects confirmed what performance measurements show: namely, that task speed was important and noticeably better for users of the help implementation where messages were displayed according to where users were. Interestingly, users also had opinions about the help activation method used. In this experiment, help messaging that answered the question "what is this/what does this do" was activated when a subject depressed the left mouse button over an interface component. While several users indicated that another method might be easier, this might preclude other help messaging. For example, an activation method that displays help whenever the cursor passes over a component might preclude including using the messaging area for another companion type of help—help that answers the question, "what do I do next" once a subject has selected (e.g., depressed and released left mouse button) over an interface component (e.g., a button). As a previous study suggests, help methods should be based on and tied to help content categories, not the other way around. That is, the taxonomy of help methods should derive from the taxonomy of help content. But issues like optimal help methods must be tested against a number of factors (e.g., industry standards, help messaging types, and so forth) and should be a subject for future work.

Additional Observations

In addition to the somewhat sterilized environment used for the experiment, routine real-world scenarios suggest that there could be significant performance differences between

"Easy to use (message help implementation in current window). Provides better input when the user needs help."

"Both (help implementations) helped complete the tasks. It was nice not to have to move the window in which I was working to get the help message."

"My reason for preferring the latter method (messages were displayed in the window where the component was located) is that you don't have to move the windows to see the help text in the main window. Maybe display the help text on the focus of the mouse rather than on left mouse click."

"Did not like the test where window covered the message window."

"I didn't like to have to push MB1 (Mouse Button 1) over the component in order to have the help message appear. It should be sufficient merely to have the mouse cursor over the object."

Fig. 6. *Selected general user comments in post-experiment questionnaire.*

the alternative help messaging implementations. Applications where dialog boxes are displayed before the application's main window appears could not benefit from messages appearing only in the application's main window. Motif's standard file selection dialog box is one common example. Placing the messages only in the application's main window also prevents users from simultaneously inquiring about two components in separate dialog boxes. This situation arises in applications where multiple dialog boxes are active during the performance of a single task. Screen clutter caused by a multiplicity of application windows on screen, too, can further complicate a user's ability to find an application's main window.

CONCLUSIONS

Other experimental studies have shown the inherent value of both user-initiated and system-initiated messaging [2], [6] strategies of application help. The central question that the current study sought to answer was this: what is the effect of advice message location on user performance? Results of this study indicate that user performance is significantly affected by the location of help messages, and is more evident as the size (and potential obscuring nature) of the secondary windows increases.

REFERENCES

[1] D. J. Mayhew, *Principles and Guidelines in Software User Interface Design.* Englewood Cliffs, NJ: Prentice-Hall, 1992, p. 569.
J. Elkerton and S. Palmiter, "Designing help systems using the GOMS model: An information retrieval evaluation," in *Proc. Human Factors Soc. 33rd Annu. Meet.*, 1989, p. 281.
[2] A. W. Roesler and S. G. McLellan, "What help do user's need? Taxonomies for on-line information needs and access methods," in *Proc. Human Factors In Computing Systems* May 7–11, 1995, pp. 437–441.
[3] W. C. Hill, "A Wizard of Oz study of advice giving and following," *Human-Computer Interaction*, vol. 8, pp. 57–81, 1993.
W. C. Hill, and J. R. Miller, "Justified advice: A semi-naturalistic study of advisory strategies," in *CHI 88 Proc.*, May 1988, pp. 185–190.
[4] J. Elkerton, "Online aiding for human-computer interface" in *Handbook of Human-Computer Interactions.* Amsterdam, The Netherrlands: Elsevier, 1988, pp. 345–364.

[5] S. K. Card, T. P. Moran, and A. Newell, *The Psychology of Human-Computer Interaction.* Lawrence Erlbaum Assoc., Inc., 1993.

[6] A. M. Furman and J. H. Spyridakis, "The effect of system-initiated advice on the use of menu navigation shortcuts," *IEEE Trans. Prof. Commun.*, vol. 35, no. 2, pp. 112–119, 1992.

S. G. McLellan received the B.A. degree in mathematics from Texas Western University, and the M.A. as well as Ph.D. degrees in english from Exeter University and the University of Texas, respectively. In 1985, he joined Schlumberger at its systems center in Austin, TX, where he manages the research and development of the information group for the center's oilfield services and products. His current research focuses on on-line information usability and methodology.

A. W. Roesler received the B.A. and M.A. degrees in mathematics from the University of Texas at Austin. He received the M.S. degree in engineering with a specialty in ocean acoustics from the Catholic University of America. As an engineering specialist with Schlumberger's engineering center in Austin, TX since 1976, he has responsiblity for developing and implementing strategies dealing with software standards, including interface design, graphics, and on-line help access.

A. L. Elliot received the B.S. degree in computer science from Georgia Institute of Technology, Atlanta, where she is now working towards the Ph.D. degree in computer science. She is a member of the Center for Information Management Research. Her current research areas include software engineering, human computer interaction, and computer-supported cooperative work.