

Social dynamics, or scientific truth, or sheer human cussedness:

Design decisions in the evolution of the Andrew user interface

Christina Haas
Information Technology Center
Carnegie Mellon University
Pittsburgh, PA 15213
(412) 268-6717

1. Introduction: Andrew's User Interface

"The Andrew Project" was an ambitious attempt to develop and deploy a prototype educational computing system for the Carnegie Mellon University campus. Andrew was originally conceived of as a system that would be usable by the entire campus community, including research and teaching faculty; graduate and undergraduate students in arts, humanities, sciences, and engineering; and administrators and support staff. The project, housed at the Information Technology Center (ITC) at Carnegie Mellon and jointly funded by IBM and CMU, began early in 1983 with the goal of building a workable campus-wide computing and communication system in five years [1].

A team of about 35 people--primarily computer scientists and engineers, but including writers, testers, graphic designers, and other professionals--worked on the project. For the bulk of the project, from 1983 to 1988, James Morris was the director of the ITC and the leader of the project; the upper management also included an associate director assigned to the site from IBM. The organization under Morris was loosely structured into small work groups under formal or informal "managers." These groups, some of which were more cohesive and focused than others, were responsible for developing various aspects of the system and included at various times one or more file system groups, a mail system group, a hardware maintenance group, networking groups, and a user interface group. Membership in groups was somewhat fluid, with changes made on the basis of manpower needs or personal preferences. Many of the members of the team had been recruited as recent computer science or computer engineering graduates from CMU, but others were experienced computer scientists or electrical engineers. Part- and full-time consultants to the project were brought in from the departments of Computer Science, Design, English, Social and Decision Sciences, and the Communication Design Center.

The users and potential users of the system were diverse and varied. This audience included developers both within the ITC and from the campus at large; administrators within the university and liaisons from IBM; students of all ages; and faculty and researchers from humanities and the arts, as well as science and engineering. Some of these user groups--particularly developers from the university and contacts at IBM-- were in fact quite vocal and had a direct impact on the development efforts. Other user groups, particularly students from most parts of the campus, had only indirect or "theoretical" impact on design decisions and, as one system designer who worked on the project put it, were a "captive, involuntary audience" for the Andrew "experiment." Some groups of students within the School of Computer Science had a more direct role, but these students were few in number and in some ways atypical because of their knowledge about

computers and their personal relationships with developers within the ITC who sometimes taught courses in CS.

The Andrew system itself [2], as well as its distributed file system [3], application toolkit [4], and message system [5] have been described elsewhere. The purpose of this paper is to describe some prototypical design decisions in the evolution of the Andrew user interface and to detail how a diverse group of developers and researchers worked to develop a computer system for an even more diverse audience. The inter-disciplinary nature of the Andrew development effort, the lack of clear consensus within the computer science community on what constitutes "good" user interface features and "good" user interface development strategies, and the multiple audiences for the system on the CMU campus--these three factors combined to make the design and implementation of the Andrew user interface particularly interesting, complex, and sometimes problematic.

In this report I describe three examples of user interface features in the Andrew system and the design decisions behind their current configurations. The three features--a window management feature that came to be called "column mode," the scroll bar, and the menus--went through several iterations as the project evolved. These three user interface features are only three among the myriad of user interface features in Andrew that underwent substantive change, but they are highlighted in this report for several reasons. First, they are ubiquitous features in the Andrew system, present in almost every application, and used on a regular basis by virtually every Andrew user. In addition, because of their pervasiveness, changes to these three features were noticeable and obvious ones to users of the system: changes to these features signalled important changes to the "look and feel" of the interface--how people thought about and characterized the system. Finally, the design decisions behind these three features illustrate the diverse ways that people can--and in fact did--work together in the development of Andrew.

This report is based on the author's own experiences as a consultant to the Information Technology Center during the development of Andrew, a member of the User Interface Group from 1984 to 1988, and a participant in many of the decisions described below. In addition, eleven other participants were interviewed in detail about their recollections of the evolution of the user interface, and these three features specifically, and their roles in the user interface decision making and development efforts. The participants in the interview study included four developers who were members of the User Interface Group, four developers who were outside this group but who were directly involved in decisions about or implementation of the features described here, the director of the Information Technology Center from 1983 to 1988, and two User Interface Group consultants (a writer and a graphic designer) who contributed to the design and redesign of the three features. Section 2 below describes and illustrates early versions and revisions of the column mode, scroll bar, and menus. Section 3 details how these revisions came about. In the conclusion I return to general issues of user interface design.

2. Interface Features: Past and Present

This section describes and illustrates, in turn, early and later versions of the column mode, scroll bar, and menus. The early versions were in use during 1983, 1984, and early 1985; the later versions (similar to current Andrew features) were begun in mid-1985 and completed between late 1985 and early 1987. It is important to realize that the three features whose evolution is chronicled here were not the only aspects of the Andrew user interface undergoing change.

Column Mode

Before mid-1985 ("pre-column mode," as it came to be called) Andrew's tiling window manager always completely filled the screen with windows for whatever processes were running. Whenever the layout changed--for instance, when windows were opened or closed--the entire

screen was redrawn and space reallocated. This procedure was slow, since redrawing takes time; confusing, since windows seemed to "grow" without the user requesting it and the algorithms for allocating space made predictions about layout hard to predict; and sometimes even silly. For instance, if all existing editors and command interpreter windows were closed, a load monitor or a clock might expand to fill the entire 19-inch display. (Of course, at the time this feature was in use, no one referred to it as "pre-column mode"--which would be a little like Cleopatra dating her letters "BC.")

Figures 1a, 1b, 1c, and 1d illustrate some typical window layouts in "pre-column mode." Note how the entire screen is filled, no matter how many windows are in use. Figure 1a shows a shell-script and a monitor; Figure 1b shows these two plus an editor. Figures 1c and 1d each show a monitor, a typescript, and 2 editing windows; however, these figures illustrate a common frustration for users with this version of the Andrew window manager: depending on the order of invoking processes and the current configuration of the screen, new windows were placed differently. This often seemed to users of the system to be "random" placement and on subsequent editing sessions they might get radically different layouts of windows. Other problems with "pre-column" mode (which are not illustrated by these figures) include that the only way a user could make more room on the screen without killing a process was to "hide" the window--so that no reminder of it was left, and that when a new window was called up, in some cases (depending on a particular screen layout) the entire screen had to be redrawn.

[Insert Figures 1a, b, c, and d about here]

In response to these problems, a change was made to the window manager in mid-1985 which resulted in what came to be called "column mode." The tiling window manager was retained, but with several changes: The screen was now divided into two vertical columns, a narrower left column, roughly one-third of the screen, and a wider right column. The width of columns could be adjusted by "dragging" the border to the left or right. Users could set preference options which place particular applications in one or the other column. Defaults place monitors and typescripts in the left column; editors, mail programs, and other applications are placed in the wider right column by default. Figures 2 a and 2b shows typical layouts with the default column mode. (Options also exist for horizontal columns and for more than 2 columns, although these have been used infrequently.)

[Insert Figures 2a and 2b about here.]

Column mode also introduced another important concept in the interface to the Andrew window manager: gray space. Gray space at the bottom of a column indicated unused space. Figures 2c and 2d show this gray space; open windows always move to the top of the column, while gray space takes up unused screen "real estate" at the bottom. The creation of new windows, then, does not usually require complete redrawing of the screen since the new window occupies what was gray space. Similarly, closing an application simply adds gray space and other windows do not expand to fill the screen as they had in pre-column mode. Even if no gray space is available, however, the creation of a new window requires only the redraw of the column in question, not of the entire screen. Gray space actually consisted of characters in a gray square font; later, ambitious users filled their own unused space with other fonts or even with whole raster pictures. A final change--"shrinking to the titlebar"-- allowed for the retaining of the title bar as a visual reminder that a process was still running even if the window was hidden; it is similar to the iconification in non-tiling window systems like the Macintosh. In Figure 2d, the 'eagle' text in the editor EZ is hidden using the "shrink to titlebar" option.

[Insert Figures 2c and 2d about here.]

The Scroll Bar

The scroll bar was used to move around and through the document. The scroll bar was also a visual indicator to show which part of the document was currently displayed and the approximate length of the document. The first version of the scroll bar was a white box with a black line to indicate the part of the document currently in view. The location of the text caret in the document was indicated by a black circle within the scroll bar; this circle would "grow" when a part of the text was selected.

Figure 3 shows an early Andrew screen with several scroll bars, labeled A, B, C, D, and E. In scroll bar A, the black line extending the length of the scroll bar shows that the entire document (in this case, mail captions) is in view; the circle at the top of the scroll bar shows that the caret or selection is at the beginning. Scroll bar B, in the body of the message, shows that approximately one-third--the middle third--of the document is in view. Scroll bar C, in the EditText window above the mail program, indicates a very long document, for here the line indicating the portion of the text in view has shrunk to a small dot. In this document, too, the caret is at the beginning--shown by the black circle. Similarly, scroll bar D is in a window with a long document, but here the typing caret is at the end of the document (because the black circle is at the bottom of the scroll bar). Scroll bar E, also in an EditText window, shows that some text is selected by the elongated circle instead of the usual small circle indicating the cursor. Here, about half of the text that is in view is selected and this is reflected in the scroll bar.

[Insert Figure 3 about here].

Two kinds of functions, indicated by two different cursors, allowed users to move through documents with this original scroll bar. The first, "thumbing," used the right pointing triangle cursor to move quickly to a general part of the document. The user would point this cursor at the area of the scroll bar representing the section of the document he or she wished to see. The second function, scrolling proper, used the up-and-down arrow cursor and allowed for more precise movements. When this cursor was placed next to a particular line of text, that line would move to the top or bottom of the screen, depending on which mouse button was used. Appendix A contains the original user documentation written to explain the use of this scroll bar. It shows an example scroll bar with these two functionally different cursors.

Changes were made to this scroll bar, beginning in mid-1985, which resulted in significant changes in the visual appearance of the scroll bar and smaller but still important changes to its functionality. In the revised version of the scroll bar (shown in Figure 4), the scroll bar itself is now gray with a white bar to indicate the portion of the document in view. The cursor location is still shown, but now with a small black bar which expands to a rectangle when something is selected. New features in this scroll bar include lighter gray "end zones" at the top and bottom of the scroll bar to represent the beginning and end of the document. This revised scroll bar uses only one cursor, the up-and-down arrow cursor. Fine, precise movements are still made by positioning this arrow next to a particular line of text and pressing the right or left mouse button to move up or down in the document. Large movements can be accomplished in two ways: one can move to the beginning or end of the document by placing the cursor in the top or bottom "end zone." Placing the cursor inside the white portion of the scroll bar, holding down the left button, and moving the mouse up or down will result in the white portion "sliding" up or down the scroll bar. Releasing the mouse button will bring the corresponding part of the document into view. Appendix A shows some early documentation describing the operation of the revised scroll bar.

[Insert Figure 4 about here].

Menus

Initially, Andrew employed hierarchical menus in which selecting certain menu items invoked submenus. Figures 5a, 5b, 5c, and 5d show examples of these hierarchical menus. There were no constraints for size built into this system, so menus could have both an infinite number of items and infinite levels of submenus. The cursor in these menus was the "pointing finger cursor" and a preference option allowed for a feature which re-selected the last selected menu item when the menus were invoked.

There were several problems associated with these menus, including unwieldiness as menu items in various applications grew and unpredictability in placement of the submenus. Figures 5a and 5b illustrate this latter problem: in Figure 5b, the menus were invoked too close to the right edge of the screen for the submenu to be placed to the right as it normally would be (and as is shown in Figure 5a). Therefore, the submenu comes up "backwards" or on the left rather than in the expected place. Figure 5c illustrates another problem: the "middle pointing finger cursor" which some users on the Carnegie Mellon campus found offensive. In addition, this cursor often came up on top of menu items. Figure 5d shows the "selection" menus (a supplementary set of menus which include style and editing commands) in the original hierarchical menu scheme.

[Insert Figures 5a, b, c, and d about here].

The Andrew menus went through several iterations; only the "final" or currently existing menus are described here. Figures 6a and 6b show currently existing menus; Figure 6a shows the "search" menu card and Figure 6b shows the "selection" menus. The revised menus (or "new menus") used a "stack of cards" metaphor to organize the commands on the menus. This immediately constrained the number of levels which the menus could accommodate to two. Commands were grouped on different cards and titles indicated the general nature of the items on a card. While holding down the mouse button(s) which invoke menus, the user moves the cursor to the left to select a particular card and then up or down to select an item on that card. Once the user moves to cards in the back, the front card or cards become gray (see Figures 6a and 6b) but by moving right the user can reactivate these cards. A "mouse hole" on the front card allowed the user to quickly reselect the last selected item. These revised menus used a straight right-pointing cursor, which appeared to the left and slightly above the first menu item on the front card.

[Insert Figures 6a and 6b about here].

3. Decision Making in the Evolution of Interface Features

Between early 1985 and early 1986 the Andrew system went through substantive changes. These changes were doubly motivated by a release of the system to the CMU campus and by an impending announcement from IBM about the IBM RT/PC--the advanced workstation upon which Andrew was to run. Improvements were made across the board--in performance, in networking, in support and documentation. Many important changes were also made to the user interface; the changes made specifically to the window layout (resulting in column mode), the scroll bar, and the menus are detailed here. Of the several changes made to the user interface, these three are in many ways representative. First, they were changes that were very obvious to users both inside and outside the ITC; second, they were changes upon which a number of people advised; finally, the decision-making behind these changes revealed a range of ways that the diverse group working on the Andrew user interface came to decisions.

The User Interface Group

The UI group was probably the largest work group at the ITC in 1985/86. It was also the most fluid, with members coming and going from the group on a regular basis. For the first several years of the project, the group had discussion leaders (including at different times the director and associate director) but no formal manager. The group was also the most interdisciplinary, with members responsible for research and testing and for writing, as well as for system design. At first, this group had less clear consensus and direction than did other groups. However, as the campus deployment and the announcement of the IBM RT/PC approached, the group (like other groups at the ITC) stepped up efforts to make significant changes to the system and began meeting more regularly than they had in the year or so previously.

A small but vocal minority of the members of the UI group were not computer scientists by training. They had been recruited from around the campus to produce documentation and training materials, to advise on visual aspects of interface decisions, and to conduct user studies. Their number ranged from three to six over the years 1985 to 1987. These "user advocates," as they characterized themselves, saw their role as advocating the needs of novice, non-technical users.

Around mid-1985, several of the "user advocates" had been involved in user testing. Recognizing numerous problems with the system, they began making frequent recommendations--some small or trivial, other more profound and sweeping--about needed improvements to the system. One computer scientist in the UI group recalled that it seemed that huge lists of problems were brought forth at every meeting: "There was no satisfying them--they kept finding more and more problems." In response to the seemingly endless list of complaints brought forth by the user advocates and the growing reluctance of the developers in the group to take these complaints seriously, the director suggested--somewhat facetiously at first--that the user advocates would be allowed "three wishes" for the immediate improvement of the Andrew interface. This suggestion was taken up and the result was a "Wish List" of interface changes which was submitted to the director and subsequently distributed to the group. This "Wish List," which included window predictability and scroll bar improvements, is included as Appendix B.

In the three sections which follow, I describe the evolution of the column mode, the scroll bar, and the menus. For each of these features, I present some of the impetus for change and who took leadership roles, the number and kind of people involved in the changes and their implementation, consensus about and acceptance of the changes at the time, and the kind of agreement about what had happened that I found among the people I interviewed after the fact (in early 1990).

Column Mode

By 1985, almost everyone within the ITC agreed that the original window layout scheme (described above) needed improvement. Because of problems in display predictability and speed, the user interface group was agreed that a change should be made. Some members of the group, including the "user advocates" who had been involved in testing, believed that predictability was the main problem; others thought the inefficiency of repainting was the main drawback of the original window layout scheme.

Early on, there was some talk of developing an overlapping window manager, but most people seemed to think this was an unlikely solution--too much effort had already gone into the tiling window manager and many people believed that overlapping window schemes can become unwieldy and "undisciplined." Some of the members of the ITC who had previously worked at Xerox began advocating a column system similar to that which had been used in the Cedar system. The early advocacy for adopting something like the Cedar columns began, not in UI meetings, but informally in the kitchen or in the office of one developer (a former Xerox employee) who wasn't in fact a member of the UI group. Many members of the group recalled that the

Director of the ITC, who had also been at Xerox during the development of the Cedar system, was a vocal advocate of the change to column mode. Several noted that advocacy on his part was somewhat infrequent: "Morris often times stayed above the fray. I mean, he was interested and he had opinions, but he didn't really push them. It was more like he just wandered into your office, asked a few questions, and went away," said one developer.

However, the column mode option soon came under discussion at the UI group meeting, where it was described, illustrated via crude drawings on a white board, and critiqued. One member of the group said, "I think most of us saw right away that this was a workable solution. But there remained lots of questions about just how the thing would look and work." At meetings and informally, members of the UI group discussed issues like: How wide should columns be? How would they be adjusted? Which column should contain which applications by default? Did it make sense to have horizontal as well as vertical columns? The concept of "gray space" to signal unused space (rather than expansion of existing windows to take up excess space) was debated, with some members of the group believing that screen space should always be utilized and others arguing that the advantages of fewer repaintings of windows were more important.

The actual implementation of the column mode happened quite simply: after several sometimes heated discussions about the needed changes, one member of the group simply "went off and did it." In the interviews conducted about this design decision (conducted almost five years after the fact) virtually everyone was in agreement that this developer had done the work, that he had done it virtually alone, and that he had done it quickly--almost overnight--and well. Even people who had initially been uncertain that column mode was a workable solution, were won over after the actual change was made: "Well, after people saw it and used it, they became convinced it was a good move." No one recalled more than one person who had ever turned column mode "off." According to one developer, "So column mode gave us some of the appearance, and some of the advantages, of an overlapping [window] system." Interestingly, the current code for defining the columns was not supposed to be permanent; the developer who worked on it saw it as a "hack" which he (and others) planned eventually to rewrite. Several people mentioned that this was only one of several supposedly temporary solutions to problems which became permanent parts of the system.

In general, although the decisions about column mode were not without controversy, there seemed to be a great deal of consensus throughout the process. Most people were convinced that something had to be done to increase window manager predictability and speed, and most recognized column mode as a workable idea in theory and embraced it as a good solution in practice. This high degree of consensus even extended to recollections about the decisions that were made: most of the people I interviewed tended to recall similar scenarios about the evolution of column mode. As will be evident in the next two sections, this sort of agreement--both during the process itself and in recalling it later--was not always the case.

Scroll Bar

In the case of the scroll bar, there seemed to have been much less consensus on the need for change. Most members of the ITC, having used no other scroll bar, "basically thought it was fine," according to one member of the UI group. The impetus for change in this case came from the user advocates who were involved in user testing. Teaching new and novice users to use the scroll bar had proven to be quite difficult. Because the two functions of the scroll bar (thumbing and scrolling) used the same narrow space, users often did one when they wanted to do the other, according to a writer who had tested her documentation. She also felt that the black line representing the document was "not intuitive," especially since this line grew shorter as the document grew longer.

In contrast to the "one man" effort which resulted in column mode, the design of the new scroll bar was the result of a small committee's efforts. About five members of the User Interface group--including two or three developers and three user advocates (a graphic designer, and two members who had done user testing) volunteered to come up with an improved scroll bar. This group began by informally surveying members of the ITC about the functions that a new scroll bar should include.

After about two weeks' work, according to one participant, this "committee" came up with at least two slightly different versions (pen-and-paper mockups) of a new scroll bar. Both versions borrowed some of their looks from the Macintosh scroll bar but duplicated most of the functionality of the original Andrew scroll bar. These versions were then introduced to and critiqued by the entire UI group. Some members made suggestions, which resulted in a third version of the scroll bar designed by the subcommittee. However, "rival" scrollbar designs were also put forth by one or more interested members of the UI group, and, according to one observer, "the scroll bar wars ensued." At the time, an "ITC T-Shirt Design Contest" was underway and one imaginative contestant put together several of the competing scroll bars in a t-shirt design commemorating the scroll bar wars.

To call these discussions about scroll bar design "wars" is somewhat exaggerated, but there were heated discussions about the looks and needed functionality of the new scroll bar. The resulting scroll bar--shown in Figure 4--combined features from several competitors in the "scroll bar wars." In fact, by 1990, no one could recall (or agree upon) which features were suggested by the small subcommittee, which came from critiques of their work by the larger UI group, and which were elements of "rival" scroll bars. Again, however, the implementation was done by one member of the group; this time, however, he basically implemented what a subcommittee had agreed upon.

Most of the people I interviewed recalled that, "once people got used to it," the new scroll bar was seen to be an improvement on the old one and quite soon replaced it in all Andrew programs. However, there was less consensus about just who had been involved in the design of the scroll bar. Some recollections were of a rather large committee that did most of the work; other people recalled that the subcommittee advised, but that most of the "hard work" on the design was done by the UI group as a whole. Still another member of the group recalled the effort as one of a single (somewhat beleaguered) individual--himself. In general, however, most people saw the effort as being one of a small committee with important input from the group as a whole and, once the design was complete, the implementation efforts primarily of one developer. However, the developer named as the implementer of the revised scroll bar by several members of the group, denied having been the one who did it. There is also a lack of consensus on whether or not the Director (who was at this time leading UI group meetings) took a stand on the scroll bar developments.

In most cases, the decisions that were made about design were understood to be decisions about "defaults" within the system. Many people within the ITC felt that "customization" was an almost sacred concept--users (in this case understood to be expert users) should always have options for how particular features within the system looked and behaved. Therefore, while certain defaults were suggested and implemented across the user interface, expert users retained the ability to override those defaults--with more or less effort. Indeed, today it is still possible to run the Andrew window manager without column mode, and a version of the original menus is still available. The scroll bar, however, is an exception to this rule. One developer who was quite active in the scroll bar design (and was in fact involved in the implementation) believed that "design is making a decision; design isn't leaving all the options open." He was one of the few advocates of disallowing preference customization of interface features. Therefore, "when I did it, I just threw out all the other [scroll bar alternatives]--they're gone." This meant of course that adoption of the new scroll bar was not the "free market" choice that using or not using some interface features entailed.

Menus

In contrast to the column mode and scroll bar development efforts which involved short periods of rather intense work, the changes to the menus "dragged on forever," as one observer of the UI group put it. And while work on the column mode and the scroll bar had for the most part been concentrated within the UI group, work on changes to the menus came from several directions and people worked on versions of the menus (and problems associated with them) simultaneously. While there had been some complaining about the original menu scheme, the work on new menus did not come out of a concerted effort by the UI group. Rather, one developer (not in the UI group) said that he "got really frustrated with the [original] menus" and, since it was currently a slow time in his own work, decided to try his hand at an improvement to the user interface. Several interview participants commented on this: they thought it underscored the fluidness of the organizational structure of the ITC that a developer essentially out of the user interface loop could be responsible for the beginnings of a major change to that interface. Most people agreed that this was a good thing: "Really there was an openness to ideas--not that people just readily accepted [the new menus] but it was seen as OK to be spending some of your time on other stuff. It essentially meant a big change for the better for Andrew."

One reason this developer met with little resistance may have been because of a similar frustration--mostly unspoken at the time--that a number of other people felt with the original menu scheme. Many people agreed that the design of the original hierarchical menu was actually a good one-- "one that a computer scientist would think was just right." The impetus for change to this scheme came about, according to one member of the group, when some developers within the ITC "began to abuse" the menu scheme, adding numerous items and levels so that the menus in some programs took up "huge amounts of space--just splat all over the screen" when they were invoked. There was also no standardization so "we couldn't get ourselves together to have the same names for the same things."

The developer who implemented the first major set of changes to the menus claims that he did a great number of versions, only a few of which were ever seen by anyone else. The idea behind the stack of cards menu which were released internally at the ITC was that "menus would always look the same--they would always have the same basic size and shape." The developer who had implemented the system said that he had been influenced by talk he had heard from some of the user advocates--"that users should develop automaticity" with computer interactions whenever possible. The stack of cards menu were an attempt to give users an opportunity to develop this automaticity. However, he also said he had done what he had been told "was basically forbidden in user interface...when you pop up the menu, the cursor actually jumps a little into place."

The revised design, the "stack of cards" menu, met with mixed reviews at first. Some people liked the change--or liked what they saw as the potential for improvement in the design--but others were skeptical. The problems with the stack of cards menu included the lack of functionality in moving back through a stack once one had passed through it, the lack of speed, and, because a feature had been added which automatically mapped old menu items onto the stack of cards scheme, sometimes very long, inconsistent, or nonsensical menu item names.

At the same time, another developer and a writer were working to make menu item names consistent and to order menu item names into sensible groups, each of which would appear on a separate card. They also decided that number of cards and number of items on a card should be limited to seven, plus or minus two, since current theories of human cognition suggest that short-term memory may be limited to this number of individual items. On individual cards, white space separated conceptually different groups of items. Some of the reticence about the stack of cards menu came from developers who had relied on lots of menu commands in their programs and were worried because the number of items that could be put on the menu was greatly diminished, or from people who thought that space between items was a "waste of screen real

estate."

By late 1985, several people from the user interface group had gotten involved in making various kinds of changes to the look and functionality of the stack of cards menus. These included changes to the menu cursor (done primarily by the graphic designer as part of a complete overhaul of Andrew cursors), the addition of a "mouse hole" (also called the "worm hole" and the "repeat spot") which quickly allowed a repetition of the previous command, and changes to make the menus paint and repaint more smoothly on the screen. Some of these changes were finally adopted into the "default" menus for campus release, while others were not, but by the January 1986 release of the IBM RT the menu scheme was "set."

Not for long, however: in the spring of that year, one member of the UI group began playing around with what came to be called "new menus." The most obvious feature change in "new menus" was that cards stacked to the left, rather than to the right, as in the original implementation. When these "new menus" were released within the ITC without warning, they met with furor. Because the stack of cards allowed an automaticity that had not been possible with the original menus, users tended to use them very quickly and to develop "muscle memory"-invoking menus and issuing commands by "feel" rather than by actually reading and selecting menu items.

The introduction of the "new menus" which laid the cards out in the opposite direction, caused confusion and a great deal of loud complaining, especially from those who had by now developed automaticity with the stack of cards menus, which included by this time almost everyone at the ITC. Users who had been able to effortlessly invoke menus and commands were now making glaring and repeated mistakes. "A lot of us felt it was a case of breaking something that didn't need to be fixed," said one member of the ITC.

It was precisely *because* users had become so accustomed to the "stack of cards" menus that their revision caused such an outraged response. According to several developers within the UI group, "once people got past being pissed," there were several advantages to these "new menus" which weren't obvious at first. Among these were faster invocation of the menus, the differentiation of the "axis of card selection" [which became left/right] and the axis of item selection [up/down], and a "hysteresis" built into the cards so that flipping to the next one inadvertently was less likely. The "new menus" at first also contained "tick" marks next to menu items which were visible on the left margin of each card when menus were invoked: these gave "users something to shoot for" but they were eventually removed by consensus since they "cluttered up the display."

In discussions in and out of meetings, in offices and halls, and via mail and bulletin boards, members of the ITC debated the relative advantages and disadvantages of the "new" vs. "stack of cards" menus. When discussions reached a seeming impasse, it was decided that a member of the UI group (one of the user advocates who did user studies) should design and run an experiment "testing" the two versions of menus, the right-stacking "stack of cards" and the left-stacking "new" menus. At first, this researcher was reluctant, believing that computer-human interaction studies can seldom show unequivocally that a particular user interface features is "best." However, she agreed to run a small study, which compared time-of-learning and error rates for novice Andrew users (incoming freshmen to CMU) and evaluated their affective responses to the two menu schemes. The menus tested combined various features under discussion and compared the two kinds of menus along six dimensions: orientation, cursor location, mouse hole location, mouse hole format, navigation mechanism and selection mechanism (See Appendix C for more information on the two menu types tested.)

The results of the study showed a slight advantage for the left-stacking "new menus," although because of the small number of subjects no significance tests were run. The most striking result of the study, however, was an order effect: subjects performed better with, and overwhelmingly preferred, the first menu type which they had learned--regardless of which one it was. The researcher concluded that *consistency* was the most important lesson to be learned from the study. The results of the study could also be used to argue that the vocal and adamant preference for the stack of cards menus over the left-stacking "new menus" was primarily because users within the ITC had learned to use this menu scheme first.

Despite the high hopes of some members of the UI group, this study did not "solve" the problem of which menu scheme was better; a decision was still necessary. By this time, the UI group was headed up by one developer who had been working on user interface issues for the two and one-half years he had been at the ITC. For the first time, there was a leader in place who had ultimate authority over interface decisions. Of course, when the director of the ITC was heading the group, he had a sort of ultimate authority but he seldom directly used it. The new manager of the UI group was jokingly called the "User Interface Czar" by members of the group, who nonetheless tried to influence him about the menu decision which was before him. The manager eventually decided--at the last minute, just days before the Fall 1986 campus release--that the "new menus" stacked to the left would be the default. When asked about this decision, the UI Czar recalled that his decision was somewhat influenced by the results of testing which showed a slight advantage for these menus. More importantly, however, were his own analysis of the two menu schemes in leading him to favor the left-stacking menus. He decided that the left-stacking new menus were primarily a "win" over the right stacking menus because of the distinguished axis of selection: left/right for card selection, up/down for item selection.

For the most part, this decision "closed the books" on menu design and redesign, although new multi-media applications in Andrew raised menus as an issue again in 1988 and the move to the X11 window manager in 1989. The evolution of the Andrew menus then was somewhat more drawn out and the benefits of the changes less clear cut than was the case with other user interface developments. The revisions to the menus also involved a large number of people, drawing on people outside the UI group and, indirectly through the novice users who participated in testing, outside the ITC/computer science community. There was also much less consensus about the process; even today there are a significant minority of people within the ITC who think the menus were "basically done wrong." The hindsight consensus about the scroll bar and column mode was much more uniform, but some people still seemed to have anger and resentment over what happened with the design and redesign of the Andrew menus. In short, while the book on the actual design of the menus may be closed, people's opinions about what happened--and their feelings about those events--certainly are not definitive or resolved.

4. Conclusions

Taken together, the design decisions described here point to several important "lessons" of user interface design. The first of these can be termed **shared responsibility**. It is clear that, in the Andrew system at least, the cooperation of a diverse group of people made the system a better one. Although ultimate authority may have rested with the director, responsibility for decisions was shared, for the most part, among any number of participants. This "shared responsibility" is evident in the key contributions to the menu design of a designer completely outside the user interface loop. Shared responsibility was also encouraged by a loose structure which allowed, or even encouraged, people to find interesting problems to work on outside their own current work scope. Further, responsibility was shared not just between developers but with people trained in complementary fields (rhetoric, graphic design, psychology) as well. Consultants from outside the computer science community, who termed themselves "user advocates," were often the impetus for attending to problems in the user interface and in some cases were instrumental in the solutions which were developed to address these problems.

The second lesson to be learned from the development of the Andrew user interface is the value of **iterative design**. Numerous versions of the user interface features described here (as well as many others) were informally tested in daily use by members of the ITC--many of whom tended to be quite vocal critics. This process of iterative development and testing tended to minimize the number of "it-just-seemed-right-to-me" decisions, decisions about interface features which were based on what just one "user" (the developer) thought made sense. Virtually every system designer who took part in these interviews acknowledged that at one time or another he had to back down and realize that what he originally thought was best solution to a problem may not have been. Again, the interdisciplinary nature of the user interface group meant that a wealth of expertise, background, and aesthetics were brought to bear on decisions. Inherent in truly open, iterative design is the notion that no feature is ever "off limits" for improvement--as we saw in the development of the scroll bar and the menus. However, as the Andrew system moved closer to becoming a "product" and as the user community grew from a small group willing to be "guinea pigs" to a larger, campus-wide group wanting to use Andrew to "get their work (teaching, learning, or research) done," the continual evolution of the system slowed considerably.

However, in some ways the more interesting, and more controversial, way to look at these interface design decisions is to ask, "But which ones were right?" By examining the three decisions described here separately, contrasting them with one another, and exploring why certain decisions were made, I would like to move beyond general "lessons" and speculate about what the Andrew experience offers in the way of specifics--about good interface design or at least about interface "success." My brief speculation will take two directions, first addressing the question "Which interface features turned out to be right?" and then discussing "What is the right way to make user interface design decisions?"

The "Right" Interface Features: Menu-driven, window-ed interfaces are now commonplace in computing across academic and business contexts, but this was certainly not the case when the Andrew project began in 1983. One way to think about the "right" interface features is to look at what kinds of features seem to have "won out" on the marketplace. The variety of menu options and styles available on widely-used computer systems suggests that questions about the "look and feel" of menus is still an open question.² Similarly, while scroll bars on commonplace in many applications, three-dimensional metaphors such as "page flipping" and "panning" continue to be explored. However, virtually all current windowing systems use overlapping windows, suggesting the strength of this design (at least for now). This strength is underscored by the decision process surrounding column mode in the Andrew system: this feature was implemented quickly, acknowledged as an improvement, modified slightly in several iterations, and used almost universally in the pre-X11 Andrew world. The ease with which the feature was designed and accepted (in contrast to the other two features discussed here which were somewhat more problematic) may be because column mode--with gray space, titlebar iconification, and user locus of control--moved the original Andrew tiling window manager closer to an overlapping window system.

The "Right" Decision-Making Process: The processes behind the decisions for column mode, scroll bar, and menus were quite distinct. In the case of pre-column mode window manager, there was a great deal of consensus on the need for a change, the discussions proceeded quite smoothly and quickly, and changes we accepted and adopted by almost everyone. The need for changes to the scroll bar were somewhat less obvious, but eventually most people agreed on the need for changes. Scroll bar changes took somewhat longer than did the column mode, involved a small subcommittee as well as the UI group at large, and the decision to adopt the new scroll bar was one of consensus, at least within the large and diverse UI group. In both of these decisions, the people who had done the work on the project--whether a couple of developers, as in the case of column mode, or a subcommittee with the help of the larger UI group, as in the case of the scroll bar--were responsible for the decisions for adopting them into the Andrew user interface. The case of menus was somewhat different: here, there remains even today (four

years after the fact) a lack of consensus on whether the menus were "right." This lack of consensus tended to concern the direction of the stacking menus (rather than the concept of stacking vs. hierarchical menus) and may have been due to the fact that while quite a large group of people--both inside and outside the UI group--worked on the menus, the decisions to deploy the left-stacking menus was one of "executive order."³ This decision-making strategy that quite rare within the ITC, at least as far as the interface was concerned, and in many ways went against the open, fluid organization that existed then at the ITC. In hindsight, the impasse about menus--created in fact by the large number of people who became involved (with vocal advocates on both sides) and exacerbated by the "failure" of "scientific truth" (in the form of the tests of menu options) --may not have been overcome in any other way than by a unilateral decision by the User Interface "Czar." However inevitable this manner of resolution may have been, it created a rift that many participants still remember and lead to a lack of consensus, and closure about this interface issue.

Acknowledgments

Special thanks to the former and current member of the Information Technology Center who generously gave time for interviews: Sandra Bond, Nathaniel Borenstein, Dan Boyarski, James Gosling, Fred Hansen, Chris Koenigsberg, Bruce Lucas, Jim Morris, Andy Palay, Bruce Sherwood, and Bob Sidebotham. Tom Neuendorffer provided a wealth of "Andrew artifacts" which proved quite useful. Pat Gera did an excellent job of deciphering tapes and typing transcripts. Nathaniel Borenstein, Andy Palay, Chris Koenigsberg, Sandra Bond, and Dan Boyarski provided insightful comments on earlier drafts. I would also like to thank Maria Wadlow and Paul Crumley for their interest in this project.

Notes

¹It should be noted that many people using Andrew today run it using the X11 window manager and so do not see column mode.

²This line of thinking was suggested to me by Chris Koenigsberg.

³Electronic mail conversations with Nathaniel Borenstein were helpful to me in developing this idea.

References

[1] Morris, J. H. *"Make or Take" Decisions in Andrew*. ITC Technical Paper #66, Carnegie Mellon University, Pittsburgh.

[2] Morris, J., Satyanarayanan, M., Conner, M., Howard, J. H., Rosenthal, D., & Smith, F. D. Andrew: A distributed personal computing environment. *Communications of the ACM*, 29 (3), 184-201, March 1986.

[3] Satyanarayanan, M., Howard, J. H., Nichols, D., Sidebotham, R., Spector, A. Z., & West, M. The ITC distributed file system: Principles and design. In *Proceedings of the 10th Symposium on Operating Systems Principles*. December, 1985.

[4] Palay, A., Hansen, W. J., Sherman, M., Wadlow, M. G., Neuendorffer, T., Stern, Z., Bader, M., & Peters, T. The Andrew Toolkit: An overview. In *Proceedings of the USENIX Winter Conference*, pages 9-21. February, 1988.

[5] Borenstein, N. S., Everhart, C., Rosenberg, J. & Stoller, A. A Multi-media Message System for Andrew. In *Proceedings of the USENIX Technical Conference*. February, 1988.

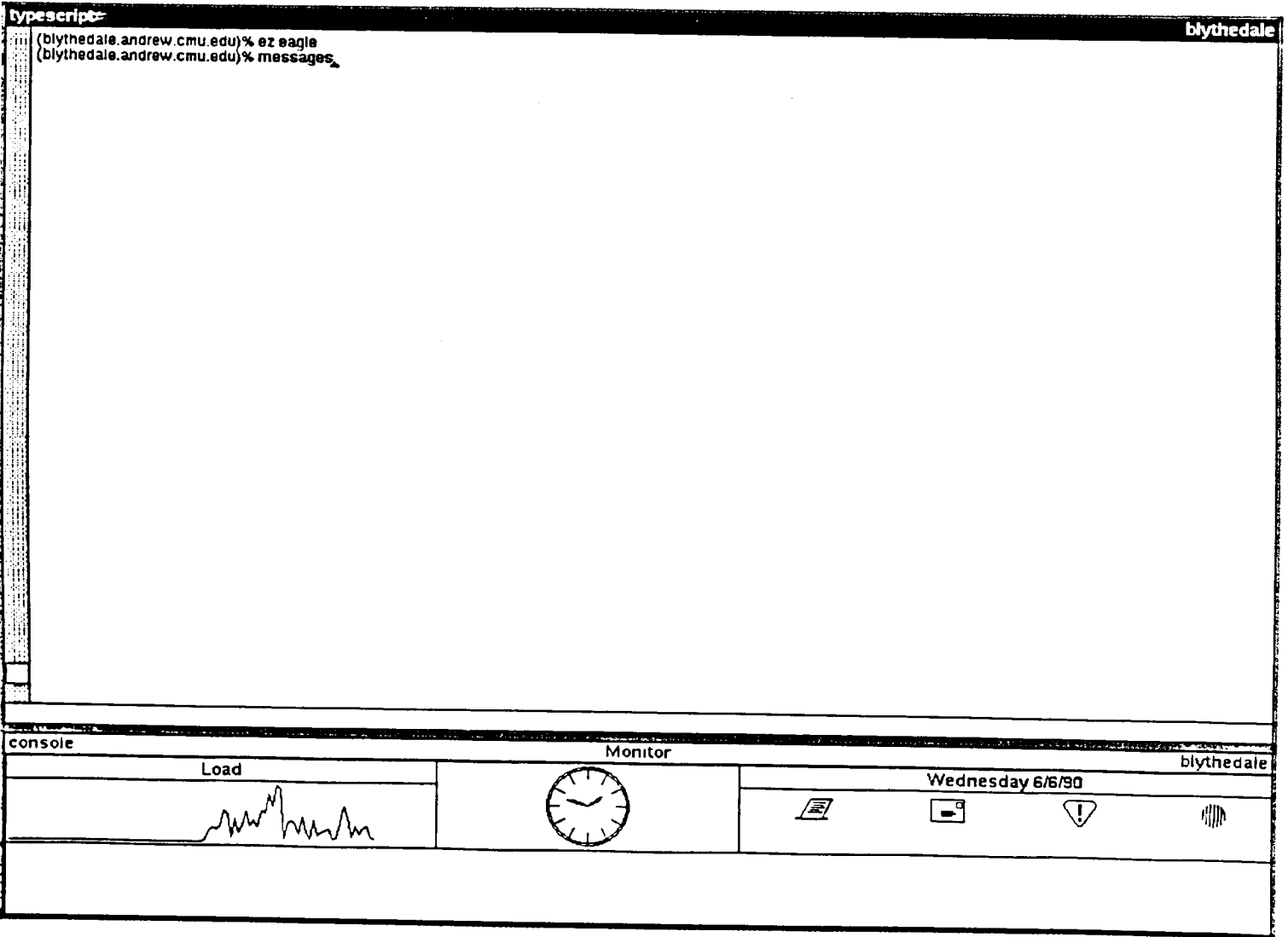


Figure 1A. Andrew "pre-column mode" window manager with two windows.

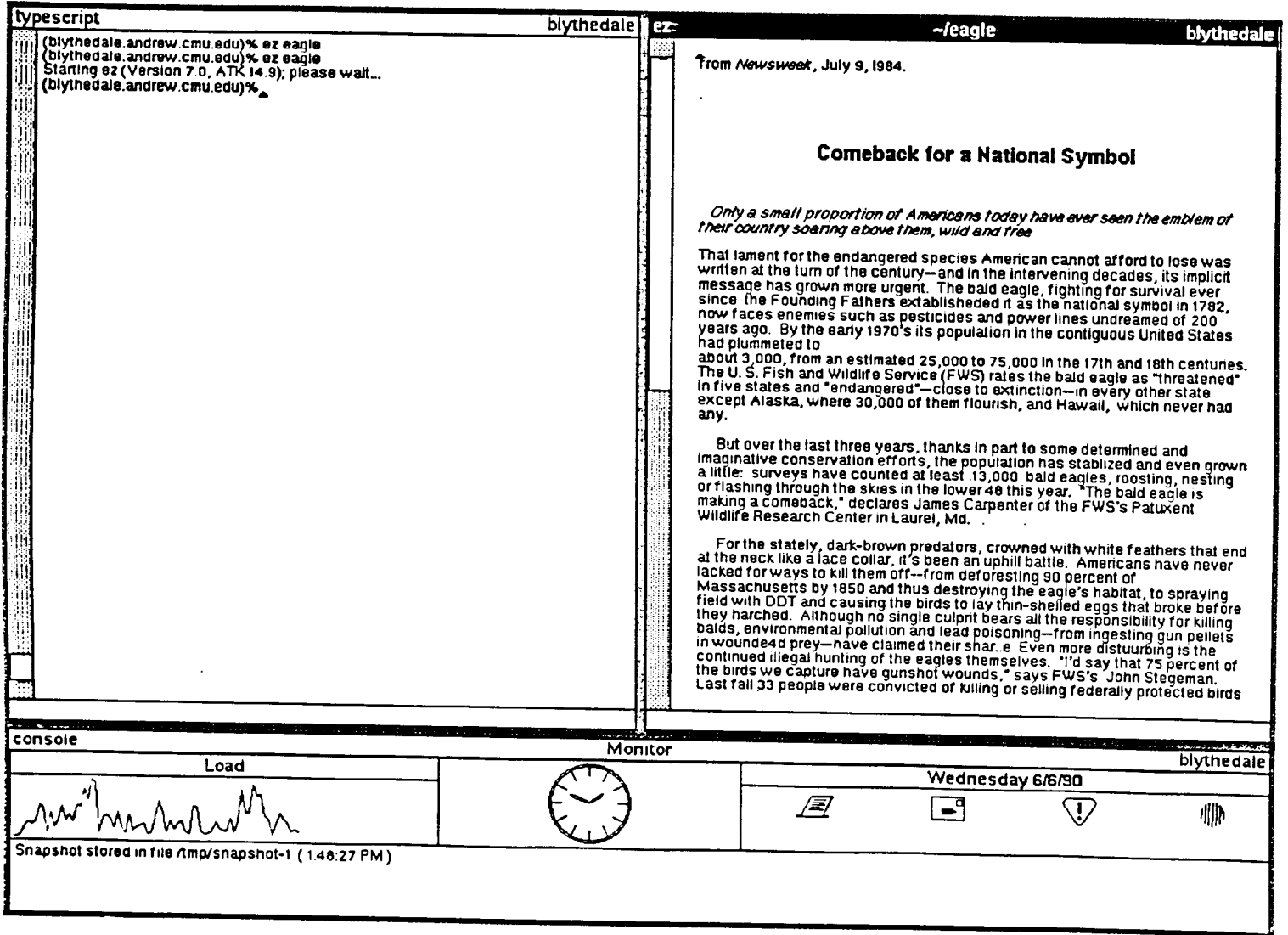


Figure 1B. Andrew "pre-column mode" window manager with three windows.

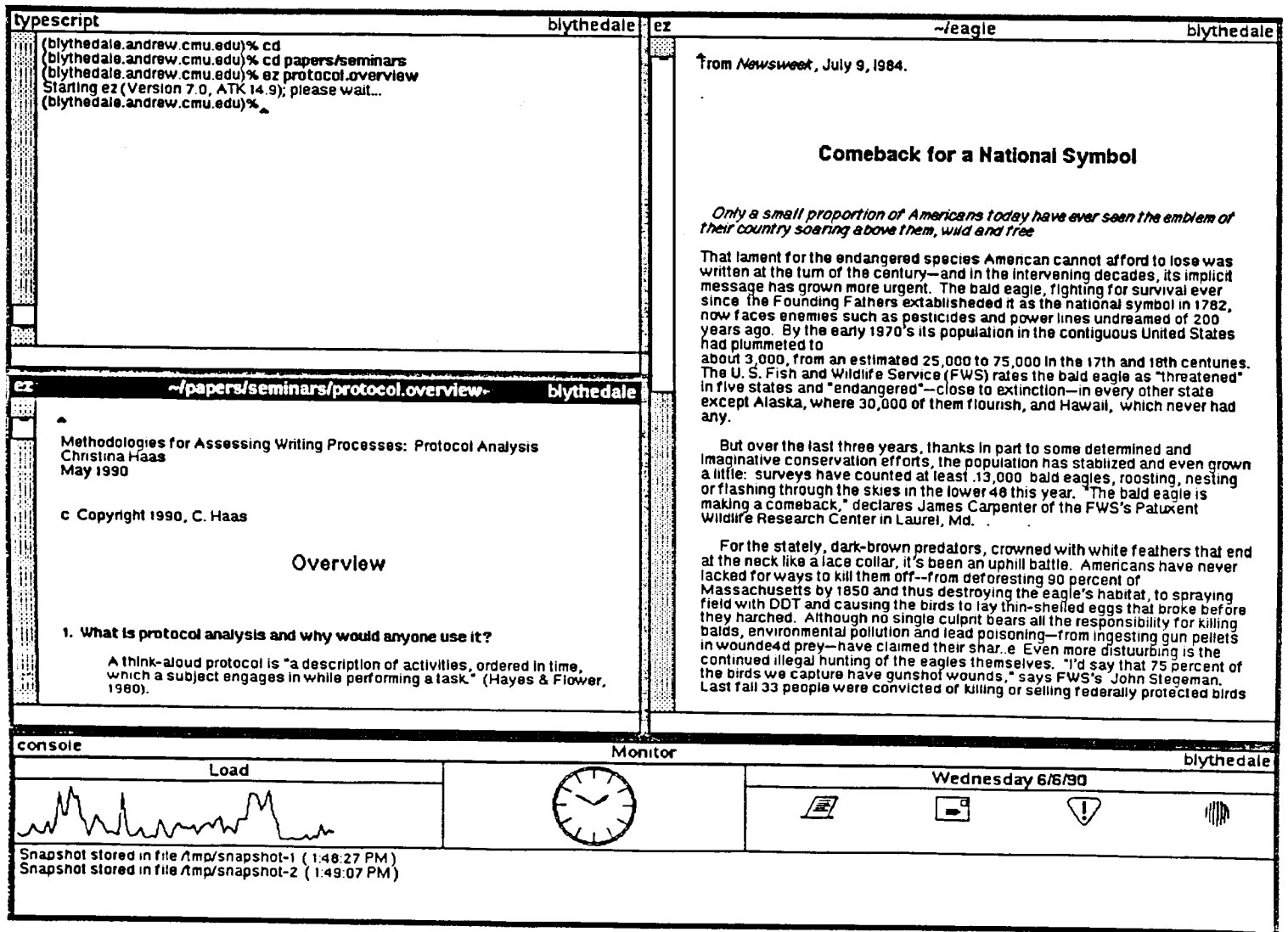


Figure 1C. Andrew "pre-column mode" window manager with four windows.

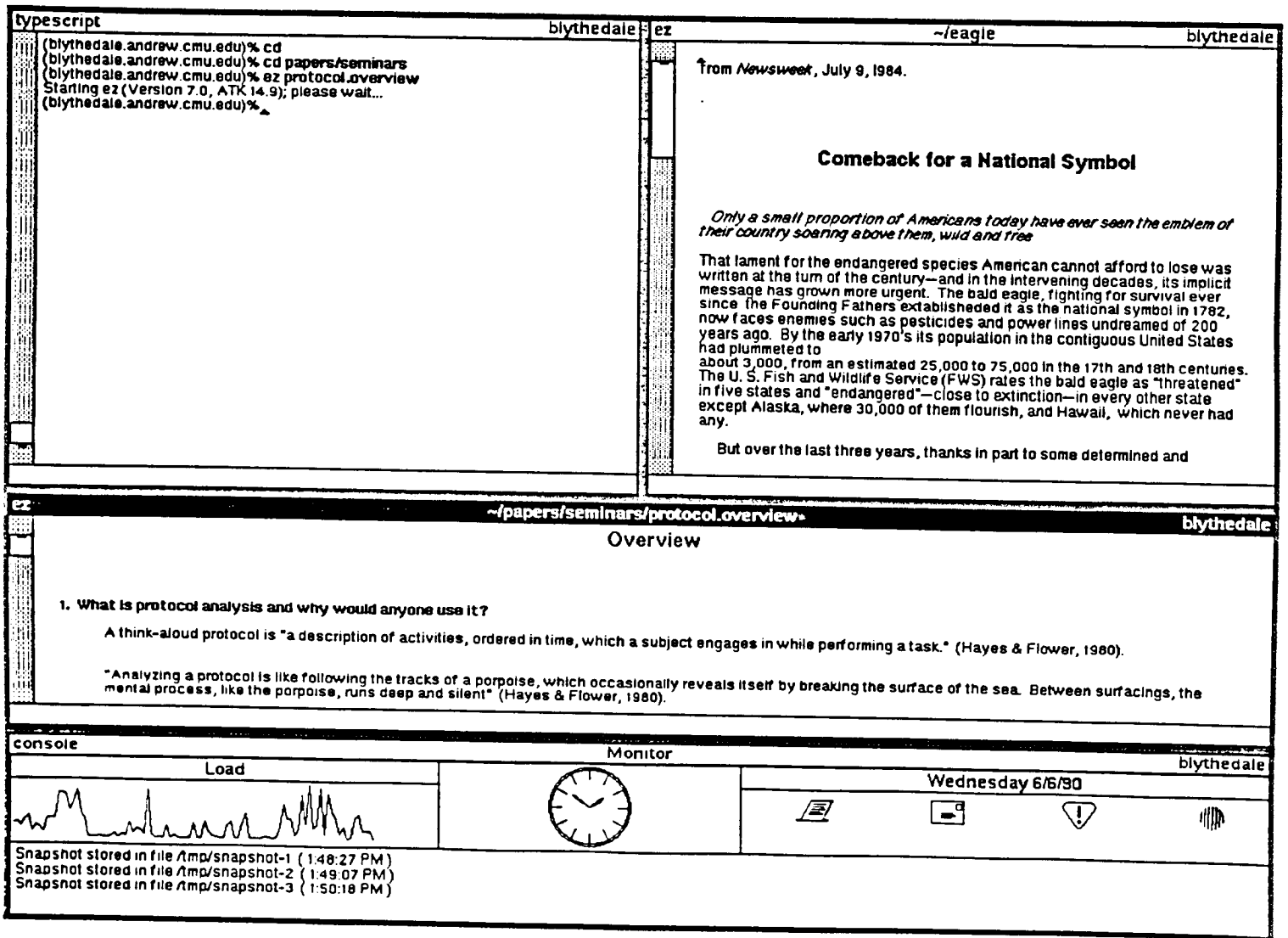


Figure 1D. Andrew "pre-column mode" window manager with four windows, alternate view.

Bypscript
(blythedale.andrew.cmu.edu)% ez eagle

Starting ez (Version 7.0, ATK 14.9); please wait...
(blythedale.andrew.cmu.edu)%

~eagle
from Newsweek, July 9, 1984.

Comeback for a National Symbol

Only a small proportion of Americans today have ever seen the emblem of their country soaring above them, wild and free.

That lament for the endangered species American cannot afford to lose was written at the turn of the century—and in the intervening decades, its implicit message has grown more urgent. The bald eagle, fighting for survival ever since the Founding Fathers established it as the national symbol in 1782, now faces enemies such as pesticides and power lines undreamed of 200 years ago. By the early 1970's its population in the contiguous United States had plummeted to about 3,000, from an estimated 25,000 in the 17th and 18th centuries. The U. S. Fish and Wildlife Service (FWS) rates the bald eagle as "threatened" in five states and "endangered"—close to extinction—in every other state except Alaska, where 30,000 of them flourish, and Hawaii, which never had any.

But over the last three years, thanks in part to some determined and imaginative conservation efforts, the population has stabilized and even grown a little: surveys have counted at least 13,000 bald eagles, roosting, nesting or flashing through the skies in the lower 48 this year. The bald eagle is making a comeback," declares James Carpenter of the FWS's Patuxent Wildlife Research Center in Laurel, Md.

For the stately, dark-brown predators, crowned with white feathers that end at the neck like a lace collar, it's been an uphill battle. Americans have never lacked for ways to kill them off—from deforesting 90 percent of Massachusetts by 1850 and thus destroying the eagle's habitat, to spraying fields with DDT and causing the birds to lay thin-shelled eggs that broke before they hatched. Although no single culprit bears all the responsibility for killing balds, environmental pollution and lead poisoning—from ingesting gun pellets in wounded prey—have claimed their share. Even more disturbing is the continued illegal hunting of the eagles themselves. "I'd say that 75 percent of the birds we captured have gunshot wounds," says FWS's John Stegeman. Last fall 33 people were convicted of killing or selling Federally protected birds including bald eagles as a result of a sting operation by FWS and the Justice Department. Many of the majestic creatures end ignominiously as headresses, rattles, and jewelry. Feather trafficking alone is believed to be responsible for the slaughter of about 300 birds a year.

To offset such assaults, conservationists are deploying an array of new repopulation techniques. The heart of these efforts is in FWS's Maryland research center, where biologists have been breeding eagles to reintroduce into areas from which the bird has vanished. Since 1977, the program has returned 70 eagles to the wild, and the success rate promises to improve with a novel strategy used for another vanishing breed. Biologists have found that peregrine falcons are likelier to hatch when incubated under a Cochlin Bartram chicken instead of in an artificial incubator. So this year, for the first time, eagle eggs lay under the fluffy white show chickens—and the just-finished breeding period produced 19 chick compared with eight last year.

Many of the eggs incubated at the Maryland research center were laid by captive eagles snatched away by biologists after five days. That doubles the eagles' productivity: within 18 to 24 days the mother lays eggs again—which it would not do if its first brood had remained in the nest. To make sure that eaglets get proper nurturing, biologists often resort to fostering. In this method, eggs laid by captive eagles are smuggled into the nests of wild eagles, who seem to have no objection to serving as adoptive parents. So far the Patuxent group has fostered eaglets to seven states.

The more common technique is hacking, a kind of outward-bound training for newly hatched birds. Research transfer eight-week-old eaglets from the breeding center to a hacking tower—a utility pole

Wrote file /at/sandrew.cmu.edu/usr16/ch/eagle;
Snapshot stored in file /tmp/snapshot-1 (2:09:44 PM)

Wednesday 8/5/90
Monitor
Load

blythedale
Console

Figure 2A. Andrew window manager with three windows in two columns.

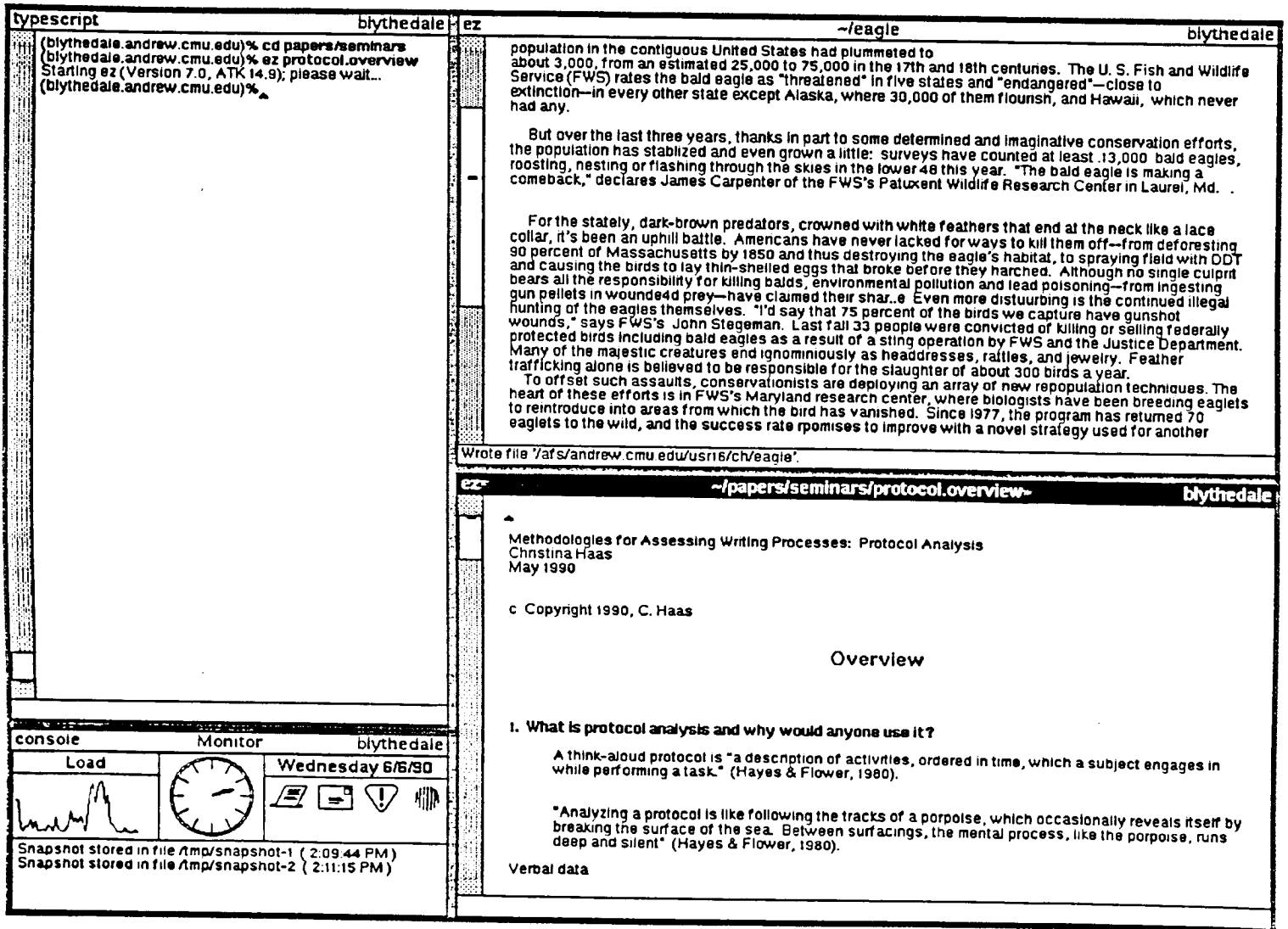


Figure 2B. Andrew window manager with two windows in each of two columns.

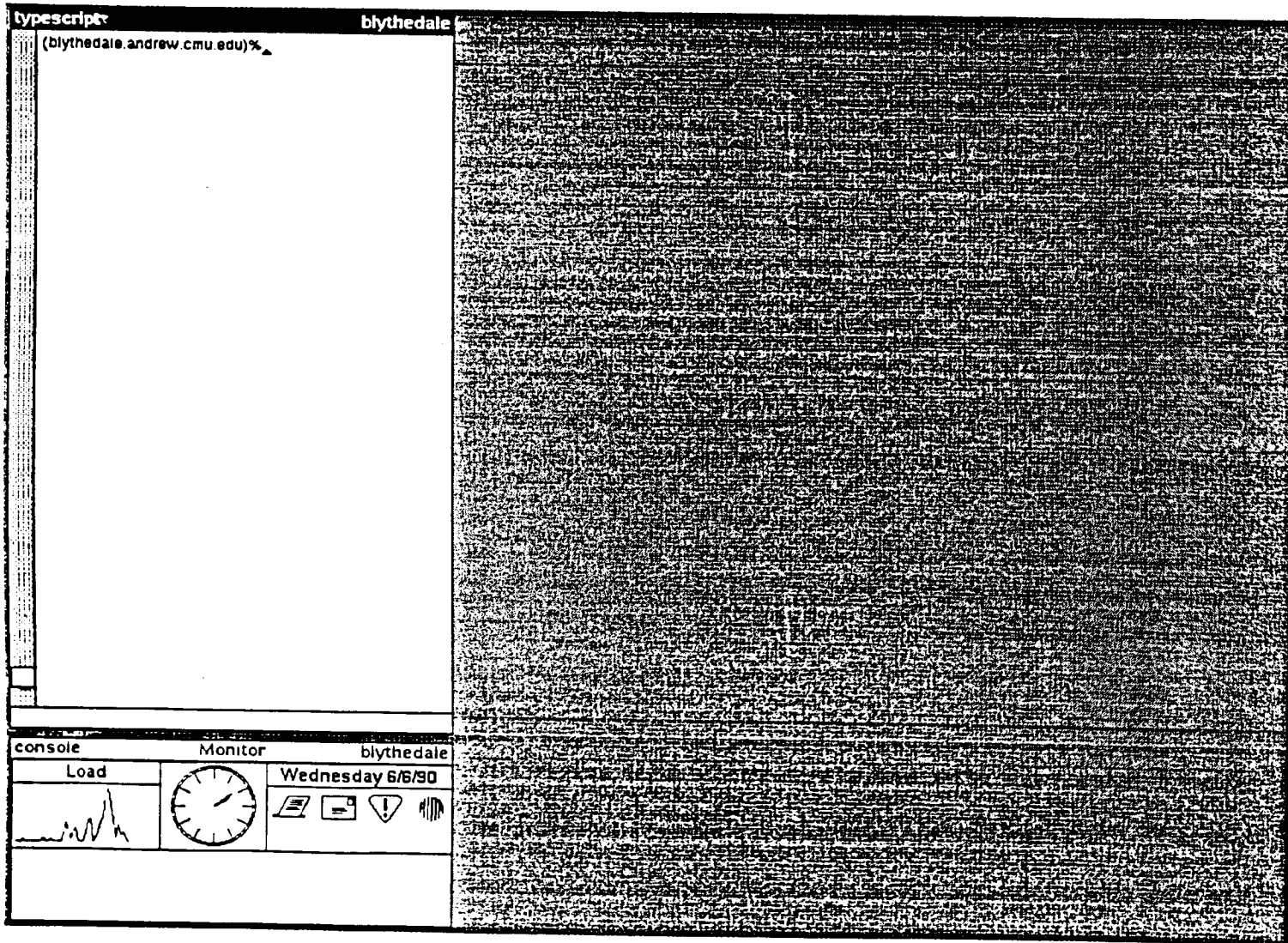


Figure 2C. Andrew window manager with two windows in left column and "gray space" in right column.

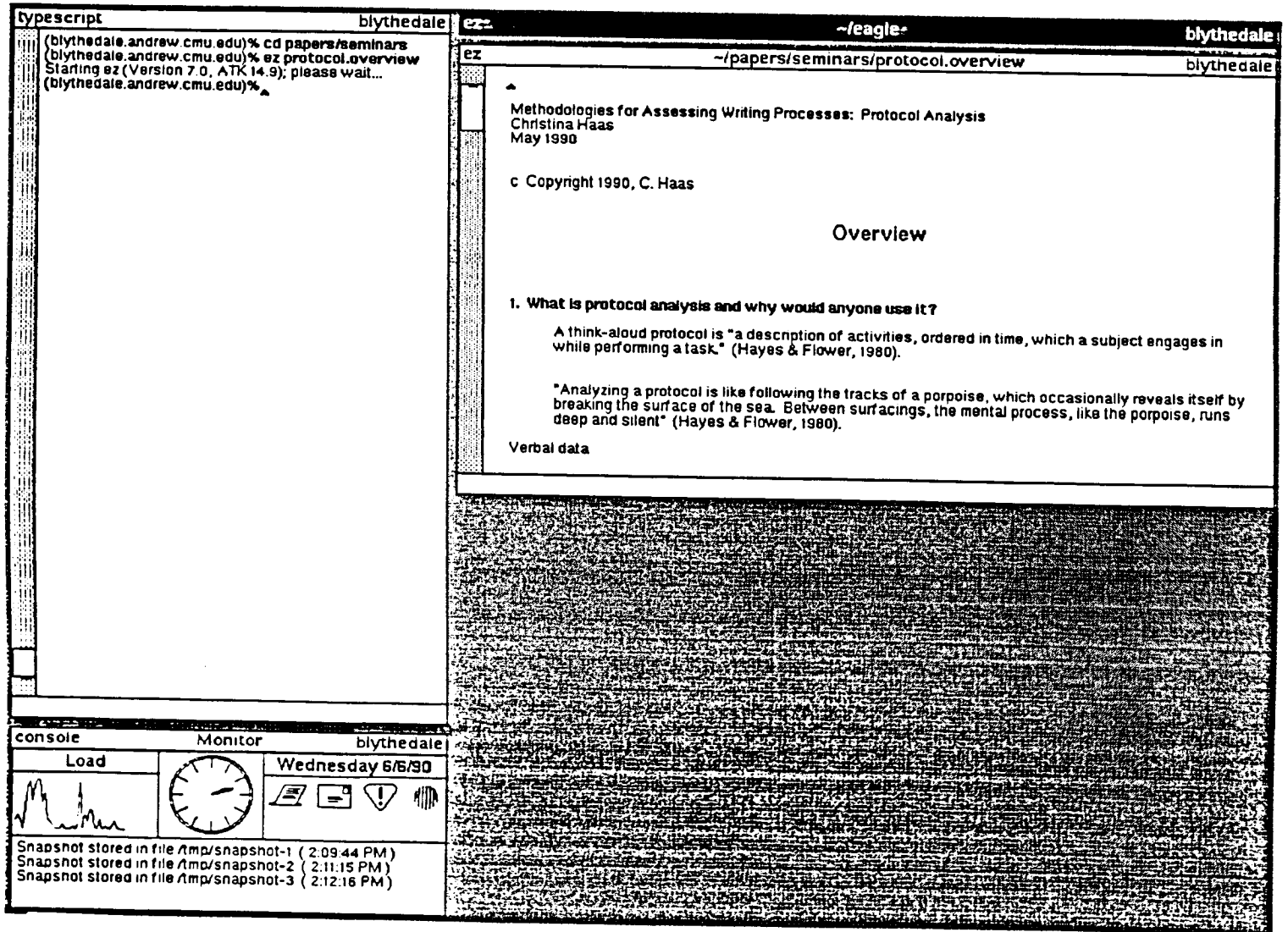
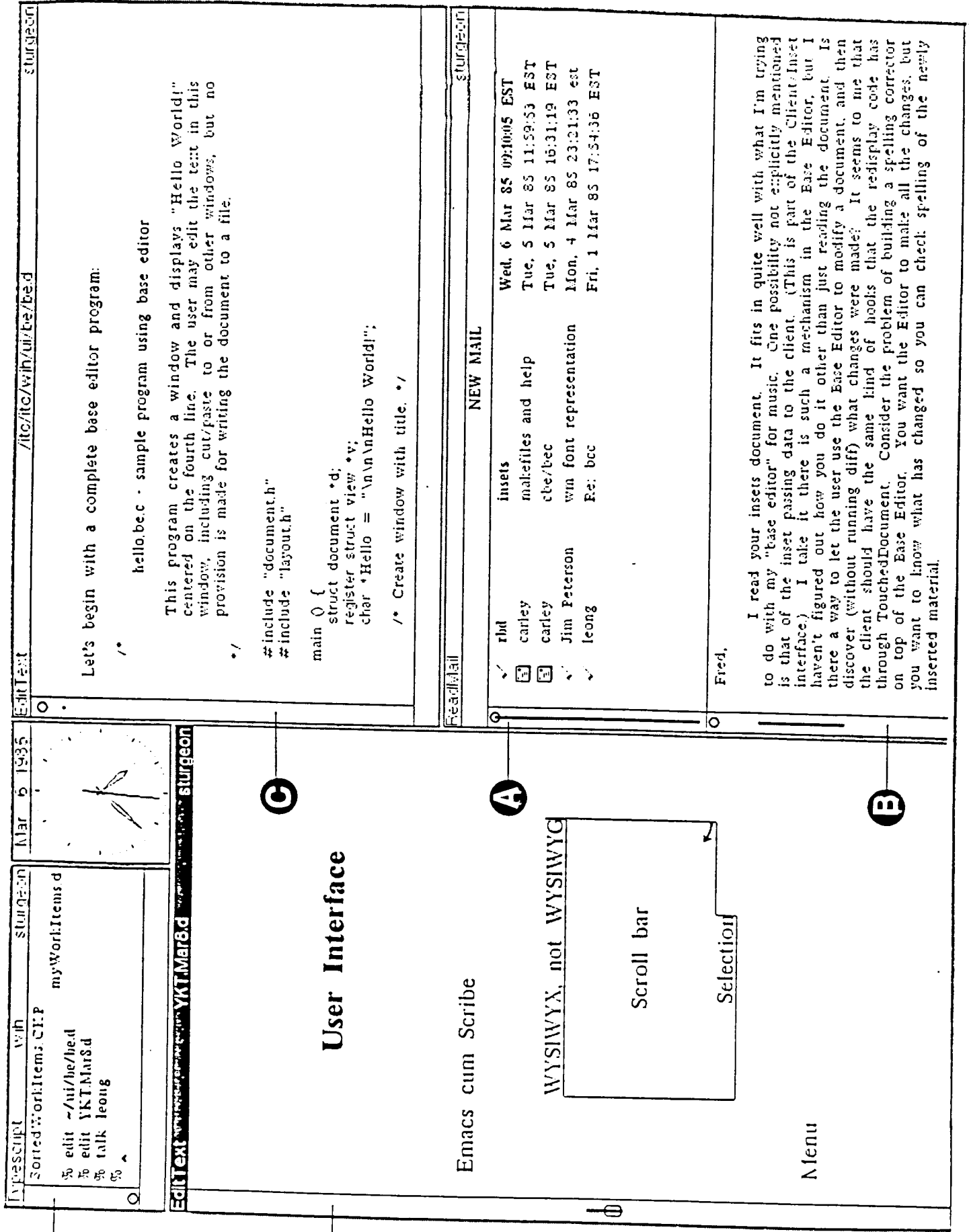


Figure 2D. Andrew window manager with two windows in left column and one open window, one window shrunk to title bar, and "gray space" in right column.



Let's begin with a complete base editor program:

```
/*
hello.be.c - sample program using base editor
```

```
This program creates a window and displays "Hello World!"
centered on the fourth line. The user may edit the text in this
window, including cut/paste to or from other windows, but no
provision is made for writing the document to a file.
```

```
*/
#include "document.h"
#include "layout.h"

main () {
  struct document *d;
  register struct view *v;
  char *Hello = "\n\nHello World!";
  /* Create window with title. */
```

Emacs cum Scribe

NEW MAIL

thd	insets	date
<input checked="" type="checkbox"/>	carley	Wed, 6 Mar 85 09:40:05 EST
<input checked="" type="checkbox"/>	makefiles and help	Tue, 5 Mar 85 11:59:53 EST
<input checked="" type="checkbox"/>	cbe/be.c	Tue, 5 Mar 85 16:31:19 EST
<input checked="" type="checkbox"/>	Jim Peterson	Mon, 4 Mar 85 23:21:33 est
<input checked="" type="checkbox"/>	leong	Fri, 1 Mar 85 17:54:36 EST

Fred.

I read your insets document. It fits in quite well with what I'm trying to do with my "base editor" for music. One possibility not explicitly mentioned is that of the inset passing data to the client. (This is part of the Client/Inset interface.) I take it there is such a mechanism in the Base Editor, but I haven't figured out how you do it other than just reading the document. Is there a way to let the user use the Base Editor to modify a document, and then discover (without running diff) what changes were made? It seems to me that the client should have the same kind of hooks that the redisplay code has through TouchedDocument. Consider the problem of building a spelling corrector on top of the Base Editor. You want the Editor to make all the changes, but you want to know what has changed so you can check spelling of the newly inserted material.

Figure 3. Early Andrew screen with original scroll bars A, B, C, D, and E.

typescript
blythedale

(blythedale.andrew.cmu.edu)% cd
 (blythedale.andrew.cmu.edu)% messages
 Starting messages (Version 7.14, ATK 14.8); please wait...

(blythedale.andrew.cmu.edu)% finger sidebotham
 Login name: bob In real life: Bob Sidebotham
 Directory: /afs/andrew.cmu.edu/usr11/bob Shell: /bin/csh
 Address mail to: bob+3@andrew.cmu.edu Affiliation: Information Technology Center
 Account used on Mon Apr 30 22:40 (6 days 11 hours ago).
 No new mail; last read Mon May 7 10:20 (4 minutes 35 seconds ago).
 No Plan.
 (blythedale.andrew.cmu.edu)%

messages -
9 Changed Folders
blythedale

mail (Mail; 0 new of 1587)
 official.andrew (Has New Messages)
 official.cmu-news (Has New Messages)
 cmu.market (Has New Messages)

3-May-90 Re: protocol analysis - Severinsson-Eklundh@nada (442)
 3-May-90 "Computers and Composition .." - "Composition Digest @vma (32643)
 4-May-90 "Computers and Composition .." - "Composition Digest @vma (19341)
 5-May-90 "Computers and Composition .." - "Composition Digest @vma (24288)
 6-May-90 Computers and Composition D.. - "Composition Digest @vma (19279)

Date: Wed, 2 May 90 08:30:23 -0400 (EDT)
 From: Nathaniel Borenstein <nsb@thumper.bellcore.com>
 To: Chris Haas <ch+@andrew.cmu.edu>
 Subject: Re: Book

Well, I hope you enjoy it, and I look forward to any comments you might have -- it is still definitely very rough around the edges, at the very least.

Checkpointing message server state... done

messages-send
Sent/Posted
blythedale

To: Nathaniel Borenstein <nsb@thumper.bellcore.com>
 Subject: Re: Book
 In-Reply-To: <kaDh-TCOM2Yt4g711S@thumper.bellcore.com>
 References: <saAoJCy00VsR0SaoIM@andrew.cmu.edu>, <EaAoVgW0M2YtACXKcC@thumper.bellcore.com>

Won't Keep Copy
Won't Clear
Won't Hide
Won't Sign
Reset

Hi Nathaniel-- I'm enjoying the book. Almost finished.

A woman from Cornell who is on a Dana Fellowship here (Nancy Kaplan) is writing an interesting piece of **why educational software fails. Her thesis (I think) is that the organizational structures of universities are not set up to foster or maintain good educ. software. She's using some pieces of great educational promise (Kaufert and Neuwirth's Comments and Notes) and arguing that the capitalist mentality of an institution like CMU is not set up to foster the essentially cooperative climate that educ. software needs.**

Anyway, I think her thinking could benefit greatly from reading *People are Perverse*. Would you mind if I copied it (or selected chapters) for her?

ch

console
Monitor
blythedale

Load Monday 5/7/90

Butler: Looking for left over processes. (10:15:16 AM)
 Butler: Cleaning out /tmp. (10:15:16 AM)
 Butler: Cleaning out /usr/tmp. (10:15:16 AM)
 Butler: Butler exiting. (10:15:36 AM)

Figure 4. Andrew screen with revised scroll bars.

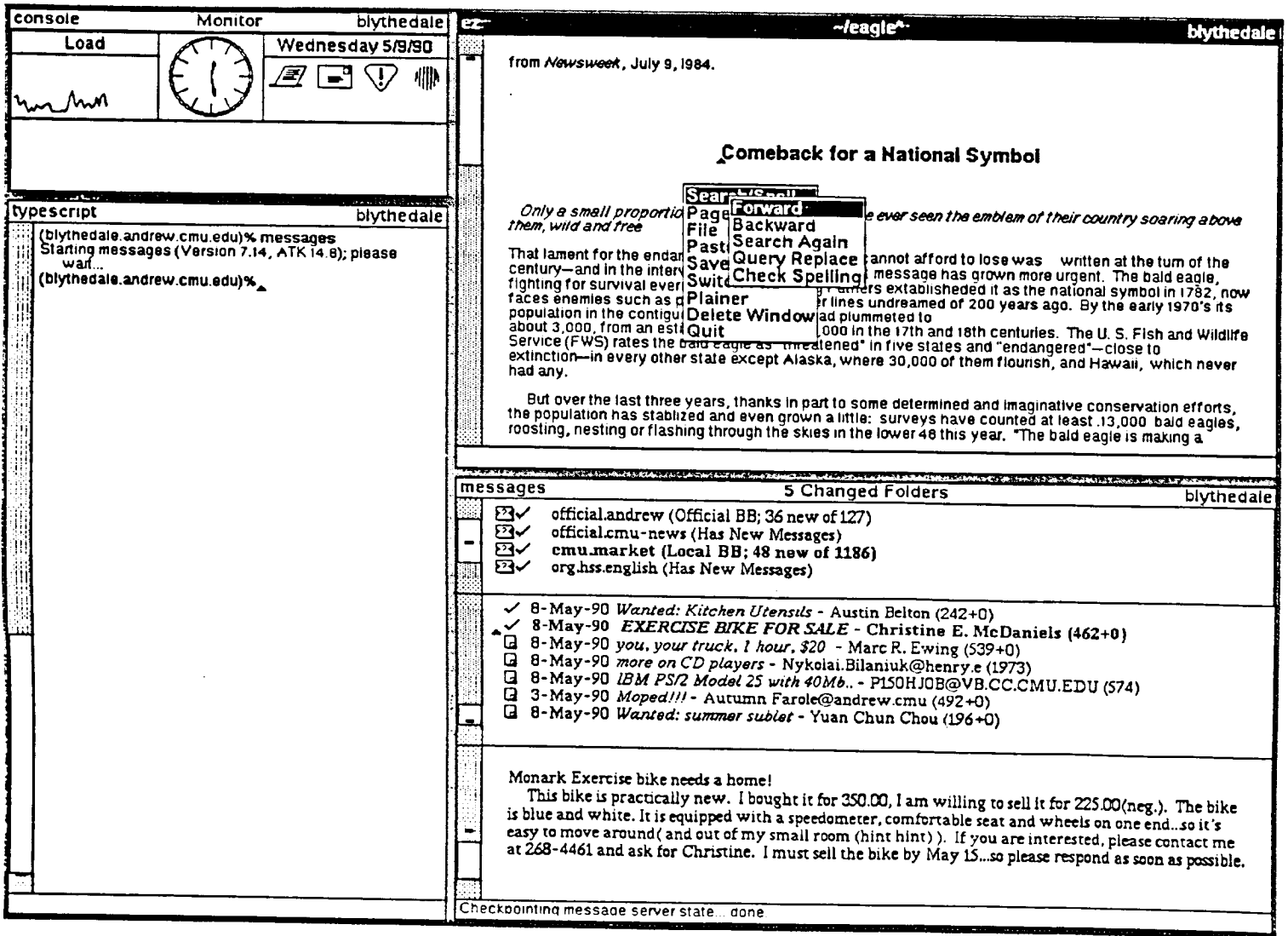


Figure 5A. Original Andrew hierarchical menus with two levels.

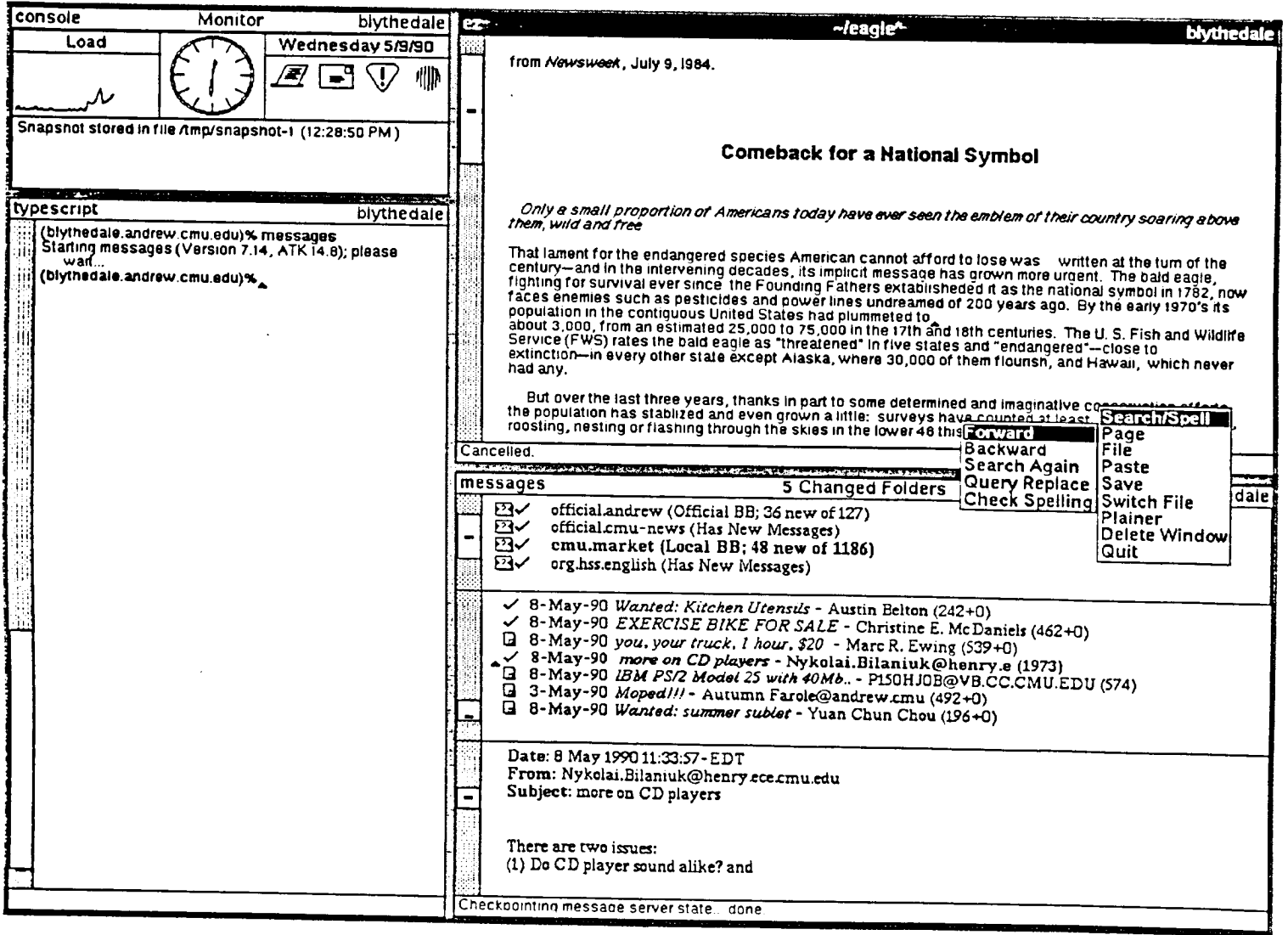


Figure 5B. Original Andrew hierarchical menus, shown with second level "backwards."

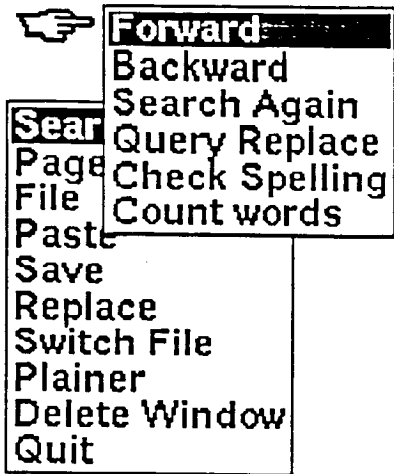


Figure 5C. Close-up of hierarchical menus with "pointing finger" cursor.

console	Monitor	blythedale	ez	-leagle*	Title MajorHeading	blythedale	
Load		Wednesday 5/9/90	from <i>Newsweek</i> , July 9, 1984.			Head Subheading Subs Chapter Index Section Bigg Subsection Bold Paragraph Italic Index Search Invisible Index	
Snapshot stored in file /tmp/snapshot-1 (12:28:50 PM) Snapshot stored in file /tmp/snapshot-2 (12:32:02 PM)		Comeback for a Nation			Font Justify Region Page File Cut Copy Save Switch File Plainier Delete Window Quit		
typescript		blythedale	<p><i>Only a small proportion of Americans today have ever seen them, wild and free.</i></p> <p>That lament for the endangered species American cannot afford to ignore—fighting for survival ever since the Founding Fathers established the nation—faces enemies such as pesticides and power lines undreamed of in the 17th century. The bald eagle population in the contiguous United States had plummeted to about 3,000, from an estimated 25,000 to 75,000 in the 17th century. The U.S. Fish and Wildlife Service (FWS) rates the bald eagle as "threatened" in five states—Alaska, where 30,000 had any.</p> <p>But over the last three years, thanks in part to some determined and imaginative conservation efforts, the population has stabilized and even grown a little: surveys have counted at least 13,000 bald eagles, roosting, nesting or flashing through the skies in the lower 48 this year. "The bald eagle is making a</p>			<p>ir country soaring above</p> <p>written at the turn of the century. The bald eagle, national symbol in 1782, now . By the early 1970's its</p> <p>The U. S. Fish and Wildlife Service (FWS) rates the bald eagle as "threatened" in five states—Alaska, where 30,000 had any.</p>	
(blythedale.andrew.cmu.edu)% messages Starting messages (Version 7.14, ATK 14.8); please wait. (blythedale.andrew.cmu.edu)%		messages 5 Changed Folders blythedale			<ul style="list-style-type: none"> <input checked="" type="checkbox"/> official.andrew (Official BB; 36 new of 127) <input checked="" type="checkbox"/> official.cmu-news (Has New Messages) <input checked="" type="checkbox"/> cmu.market (Local BB; 48 new of 1186) <input checked="" type="checkbox"/> org.hss.english (Has New Messages) <ul style="list-style-type: none"> <input checked="" type="checkbox"/> 8-May-90 <i>Wanted: Kitchen Utensils</i> - Austin Belton (242+0) <input checked="" type="checkbox"/> 8-May-90 <i>EXERCISE BIKE FOR SALE</i> - Christine E. McDaniels (462+0) <input checked="" type="checkbox"/> 8-May-90 <i>you, your truck, 1 hour, \$20</i> - Marc R. Ewing (539+0) <input checked="" type="checkbox"/> 8-May-90 <i>more on CD players</i> - Nykolai.Bilaniuk@henry.e (1973) <input type="checkbox"/> 8-May-90 <i>IBM PS/2 Model 25 with 40Mb.</i> - P150HJOB@VB.CC.CMU.EDU (574) <input type="checkbox"/> 3-May-90 <i>Moped!!!</i> - Autumn Farole@andrew.cmu (492+0) <input type="checkbox"/> 8-May-90 <i>Wanted: summer sublet</i> - Yuan Chun Chou (196+0) <p>Date: 8 May 1990 11:33:57 - EDT From: Nykolai.Bilaniuk@henry.ece.cmu.edu Subject: more on CD players</p> <p>There are two issues: (1) Do CD player sound alike? and</p>		
Checkpointing message server state... done.							

Figure 5D. Original Andrew hierachical selection menus.

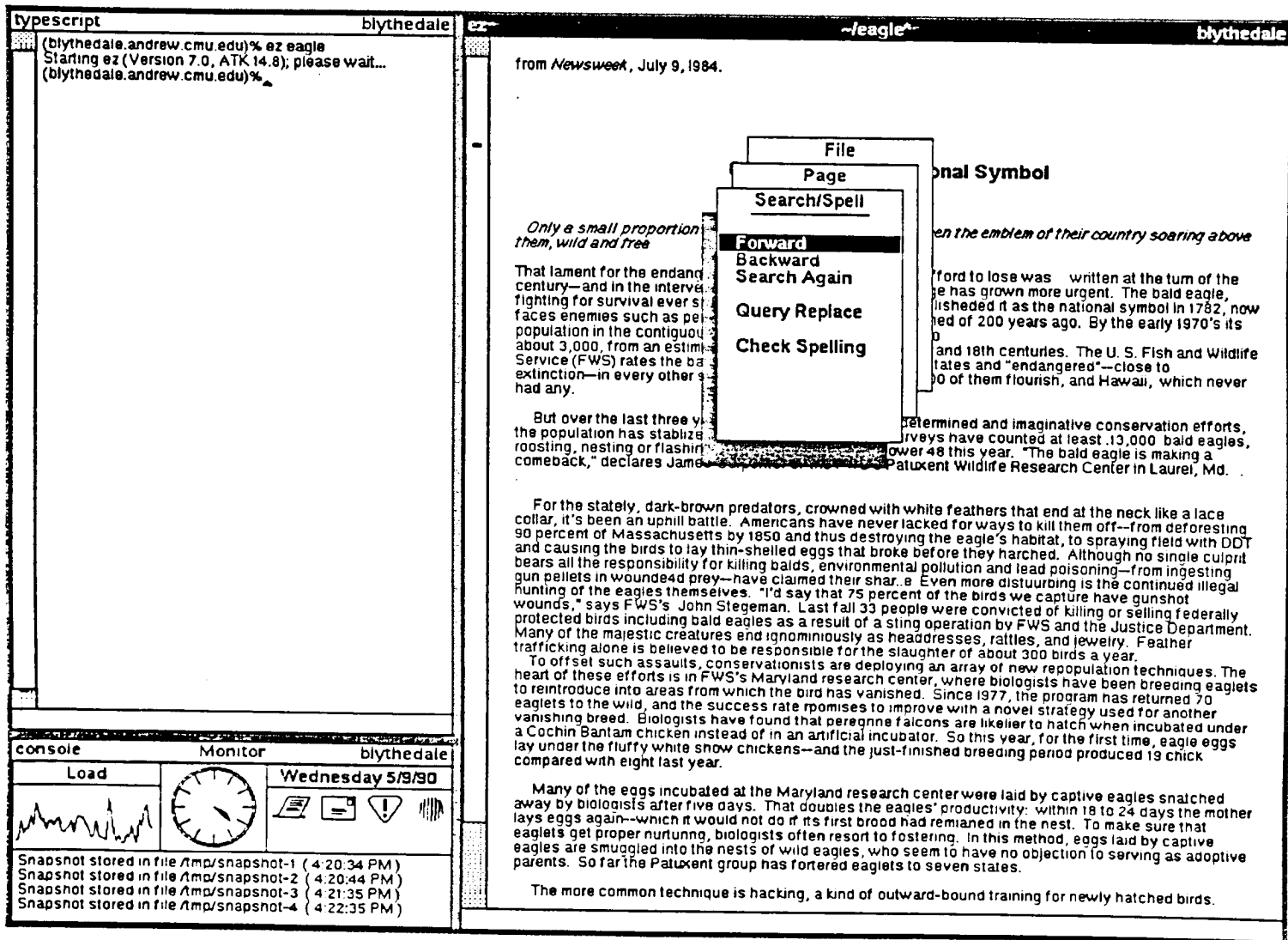


Figure 6A. Revised "stack of cards" menus.

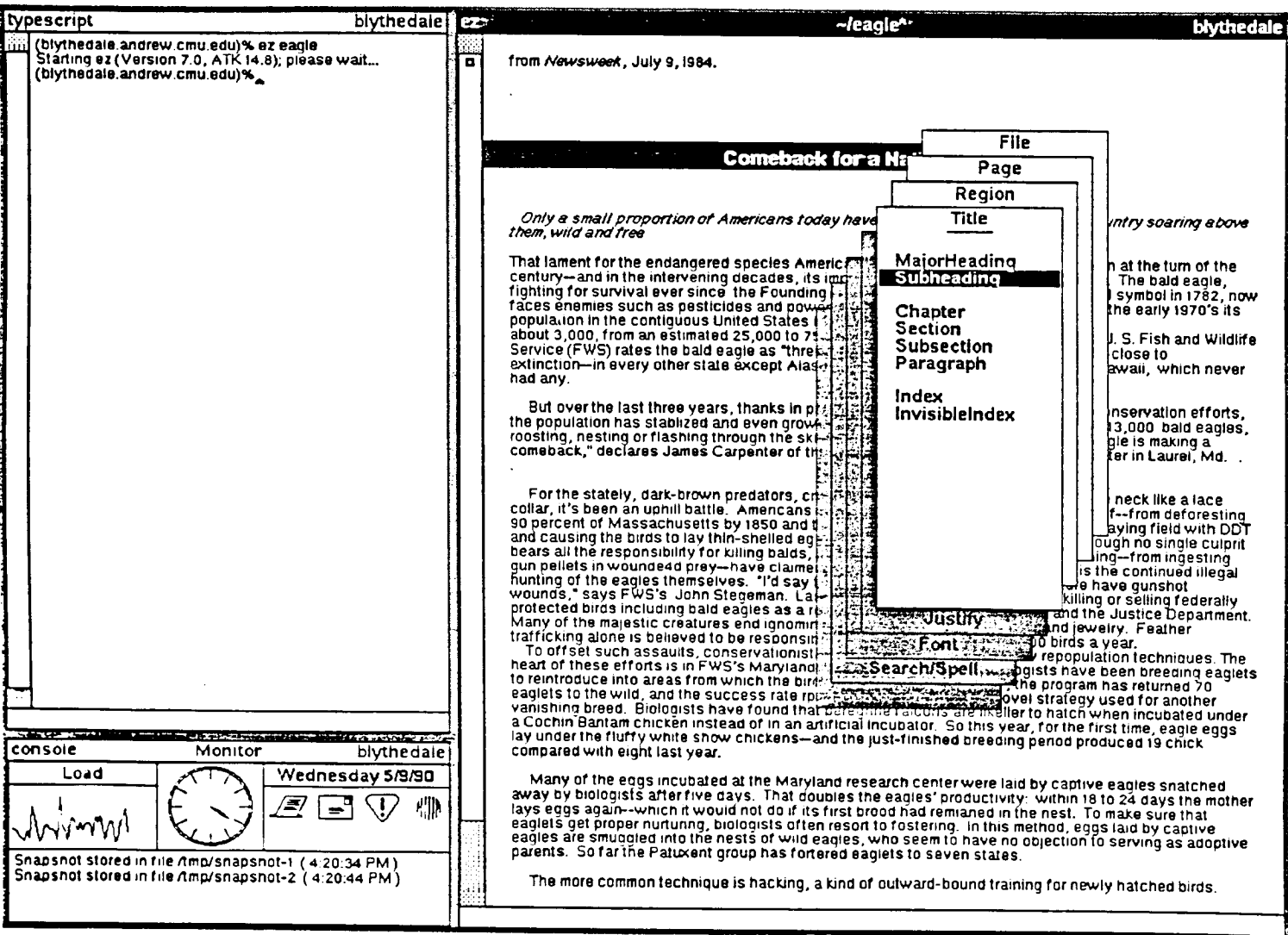


Figure 6B. Revised "stack of cards" selection menus.

Appendix A

Moving through a Document

The scroll bar is the column on the left of any window containing information that may be too long to fit entirely in a window, for example, Typescript. When a document is too long to fit entirely in a window, you can use the scroll bar to bring other parts of the document into view.

The scroll bar represents the entire document. The striped area at the top of the column represents the beginning of the document; the striped area at the bottom, the end of the document. The white bar represents that portion of the document currently in the window. If the document is very long, the white bar will be small, because only a small portion of the document will be displayed in the window. If the document is very short, the white bar will run the length of the scroll bar, indicating that the entire document is in view.

The caret inside the scroll bar indicates the location of the text caret in the document. When you use the scroll bar to move to another part of the document, the text caret will remain where it was before you moved. To move the text caret to the part of the document now in view, simply click the left button where you want the text caret to appear. The text caret will move to the place you clicked

Making large moves with the left button

Move to the beginning of the document: Click the left button in the top striped area.

Move quickly to another part of the document: Hold down the left button anywhere on the white bar and slide up or down. When you release the button, the corresponding part of the document will come into view.

Move to the end of the document: Click the left button in the bottom striped area.

Making smaller moves with the left and right button

Move a line of text to the top of the window: Position the cursor inside the scroll bar next to the line of text you want to move to the top and click the *left* button.

Move the top line of text in a window next to the cursor: Position the cursor inside the scroll bar next to where you want the text to move and click the *right* button.

Strategies for using the scroll bar

To move a screenful at a time toward the beginning of the document:

Place the cursor near but not in the bottom striped area and click the *right* button.

To move a line at a time toward the beginning of the document:

Place the cursor opposite the second line on the screen (but not in the striped area) and click the *right* button.

To move a screenful at a time toward the end of the document:

Place the cursor near but not in the bottom striped area and click the *left* button.

To move a line at a time toward the end of the document:

Place the cursor opposite the second line on the screen (but not in the striped area) and click the *left* button.

Appendix B

June 22, 1985

As Jim noted, the people who designed the STAR interface *began* with the fundamental interface principles, and devoted 30 work-years to the design. Obviously, we can't do this, but we can borrow another important principle that the designers at Xerox used and apply it after the fact, as it were: *task analysis*. Task analyses or "scenarios" of users (including descriptions of the users, their needs, their typical tasks and goals and methods) could be developed by members of the interface group. (Connor tried to get the Scholar's Workbench group to do something similar.) We might draw upon the survey that Chris Koenigsberg is doing now and the one that the CDC group assisted with last spring.

Now for my three wishes...

First, we need to make the window placement consistent. Not only do particular functions need to consistently appear in particular places (I would say as defaults, with the users able to change them if they want), but some functions--EditText for certain, maybe other editors and Mail and News--should appear in a 9 1/2" by 9 1/2" window. (The tests I ran last summer show this to be most efficient.) Further, I think for the October deployment certain windows should come up as defaults, in set places and with set sizes. I am going to contact as many of the users of Andrew (within and outside the ITC) to talk to them about the number of windows they typically have up, what they are, etc., to help determine what these default placements should be.

Second, error message and system message must be cleaned up. Even after over a year on this system my stomach still jumps when messages (from Vice?) cause the window manager to scroll up. We've been told for a long time that some of the more horrendous error messages will "eventually go away." I think they should go away by October.

System status message within the interface are also weak. The hourglass that goes away prematurely is but one annoying example, and the only way I can be sure that a document is on its way to "print" is to say "abra cadabra" three times after I give the command. Since Sandra and I for the most part don't know what the messages mean, we can't clean them up. If someone would agree to sit down with us for a half a day, or a day, or two days, we could systematically go through and make the messages specific, constructive, positive, and consistent. People who have deployed machines now would also be helpful in this area.

Third, the scroll bar remains problematic. It's difficult to use because it's so narrow, and some people have reported that the way it operates seems counterintuitive. The subjects I trained on the system last summer were enthusiastic about it, although their only basis of comparison was with control keys. It did take some time to train them on it, and while they seemed to *prefer* the scroll bar, their performance was not any better with it. I think we should explore the possibility of borrowing the Mac scroll bar, and at the very least the default should be a wider scroll bar (with the suitable provisions for hacker tailorability). Since Sandra and I will both soon run subjects, we can keep track of scroll bar training time and problems our subjects have in using it. I might also be able to design some simple experiments to test versions of the scroll bar.

Another valuable change to the scroll bar would be (this was originally Fred's idea) indicator marks to show section breaks (and/or hard copy pages) for long documents. The work I have done with computer writers shows that people tend to get lost in long documents on line since they don't have the physical and spatial cues that they have with paper. Charlie Wiescha's work with conceptual models also shows that graphic representations or diagrams help people get a sense of the model they are working with. Both of these results seem to indicate that people might be helped by a more detailed scroll bar. Graphic indicators for page or section breaks might help to alleviate problems in moving through texts and in getting a sense of the structure of documents. Such marks could be implemented either on the current scroll bar, or on the Mac-like one if we implement it. This would be easily transferred to ZIT where it might be especially useful.

The *less* arbitrary these design decisions the better, and although we don't have time to do experimental tests on them for the next deployment, we can tap the vast store of information that current users of the system (outside and inside the ITC) can provide. I am planning to interview as many current (outside) users as I can in the next few weeks about the three important areas I have outlined above. We could also implement different versions--of the scroll bar, say--internally and see how people here react.

Finally, I'd like to comment on the need for a handsome, friendly, even slick interface: certainly an attractive interface will help to sell machines. But I think there's a couple of other things we should also keep in mind: first, an attractive interface is a good PR move. I think sometimes here at the ITC we forget how closely we are being watched--by this university community as well as the "real world." Like it or not, people will judge the success of our efforts on how Andrew *looks*, as well as how he performs--and I think this is true for people in CS as well as people in Design and Psych.

Another reason for doing our best to make Andrew as handsome and friendly as possible has to do with a certain approach to computer use that I think it behooves us to adopt at an educational institution. Computers should be available and inviting for *everyone* on this campus. We pay lip service to this idea, but saying that "people who can't work around our interface can just use the Mac or the PC" (a quote from last Thursday's meeting) exhibits an elitist attitude that is inappropriate. This system is exciting and powerful; using it will make work and learning more exciting and efficient and fun for everyone. Users should not have to prove themselves "worthy" (by either their doggedness or their prior experience) of enjoying this technology. Andrew should be used by everyone and I see part of my work here as trying to make the case that it can and should be.

Novice Performance and Evaluation of Menu Schemes

Vincent Rago and Chris Haas

This paper describes the findings of tests run to determine which of two menu schemes was better suited as a default menu for the Andrew System. Data were collected on six novice subjects' performances and preferences concerning the two menu schemes. It was concluded that performance was independent of preference and that an order/performance dependency existed (an example of the importance of consistency in a user interface).

INTRODUCTION

Motivation

Pop-up menus are an integral and central part of the Andrew system's interface. As such, it is important that the menu scheme provided for users be as easy to learn and use as possible. The focus of this study is to find out what available menu characteristics should be included in the default menu scheme provided to novice users who receive an Andrew account.

Hypothesis

We expect differences in novice subject performance and subjective preferences between menu schemes that have different features. Performance will be better with less complicated menu schemes and menus that provide visual cues. Also, these menus will be received more favorably.

METHOD

To study the hypothesis, subjects were observed and tape recorded while completing two task sessions. To allow for unobtrusive observation of menu use, these sessions were described as an orientation to Andrew. In each session the subject was given a tutorial, taken from Modules 1 and 2 of the Andrew User's Guide, that contained thirty menu selection tasks. Subjects were told that they were being observed and recorded, but no indication was given as to what part of their actions was being studied.

Each subject used two menu schemes, one menu in the first session, the other in the second. The two menu schemes were counter-balanced across the sessions to control for order and task dependency. After the first session, the subject left the room and the menus were changed. The subject then returned, completed the second session, and answered questions about the sessions.

Variables

Independent

The independent variables are features of menus. Because of this, two menu schemes that differed in as many of the options as possible were used. The two menus used were StartOutside and NewMenus. Examples of the menus can be seen in Figures 1a (StartOutside) and 1b (NewMenus), detailed descriptions of the menus can be found in Appendix A. The independent variables were operationalized and coded as follows:

- Orientation: Menu cards are laid either left to right (StartOutside) or right to left (NewMenus).
- Initial cursor location: Each menu option has its own cursor location. In this study, StartOutside's cursor is found to the left and outside of the first menu card. NewMenus' cursor is found to the right and slightly lower than the upper left hand corner of the first menu card.
- Repeat spot location: StartOutside's is outside and to the left of the upper left hand corner of the first menu card, directly underneath the tail of the arrow in its initial position. NewMenus' is located in the center of the first menu card above the first option.
- Repeat spot format: StartOutside's is a square with rounded edges about the size of a character. NewMenus' is a bullseye shape.
- Navigation mechanism: To navigate through the menu cards in StartOutside, the arrow cursor is placed in the exposed region of the menu cards not active. Cards initially preceding the current active card are shaded as a visual cue. To navigate through NewMenus right to left, the arrow cursor is placed in the exposed region of the menu cards not active. To navigate from left to right the arrow cursor is placed over the region of the menu that overlaps the menu beneath and to the right. NewMenus uses no shading.
- Selection mechanism: To highlight a selection in StartOutside the arrow cursor is placed in the region defined horizontally by the edges of the menu card and vertically by the character line containing the intended selection. To highlight a selection in NewMenus the arrow cursor is placed in the region defined horizontally from the left edge of the menu card to the left edge of the text and vertically by the character line containing the intended selection. *EXCEPTION: If the first menu is the current menu the selection region is defined horizontally by the edges of the menu card.*

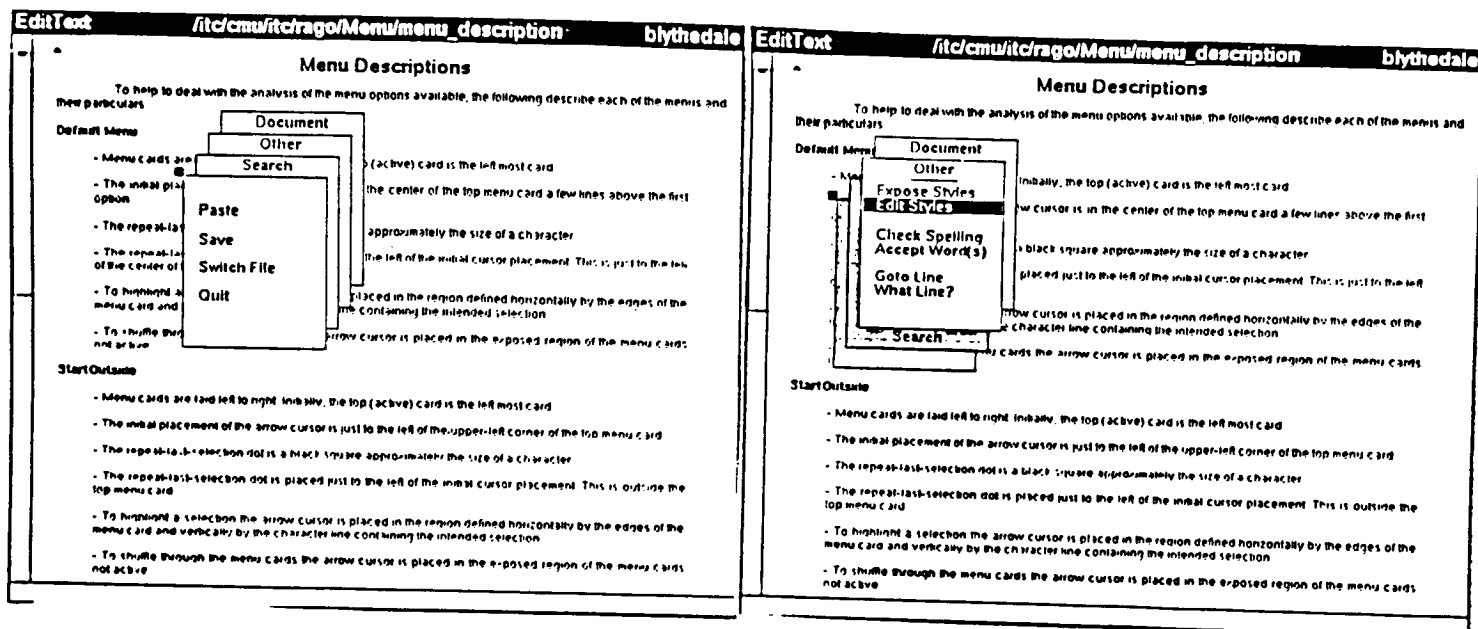


FIGURE 1a: An example of StartOutside menus

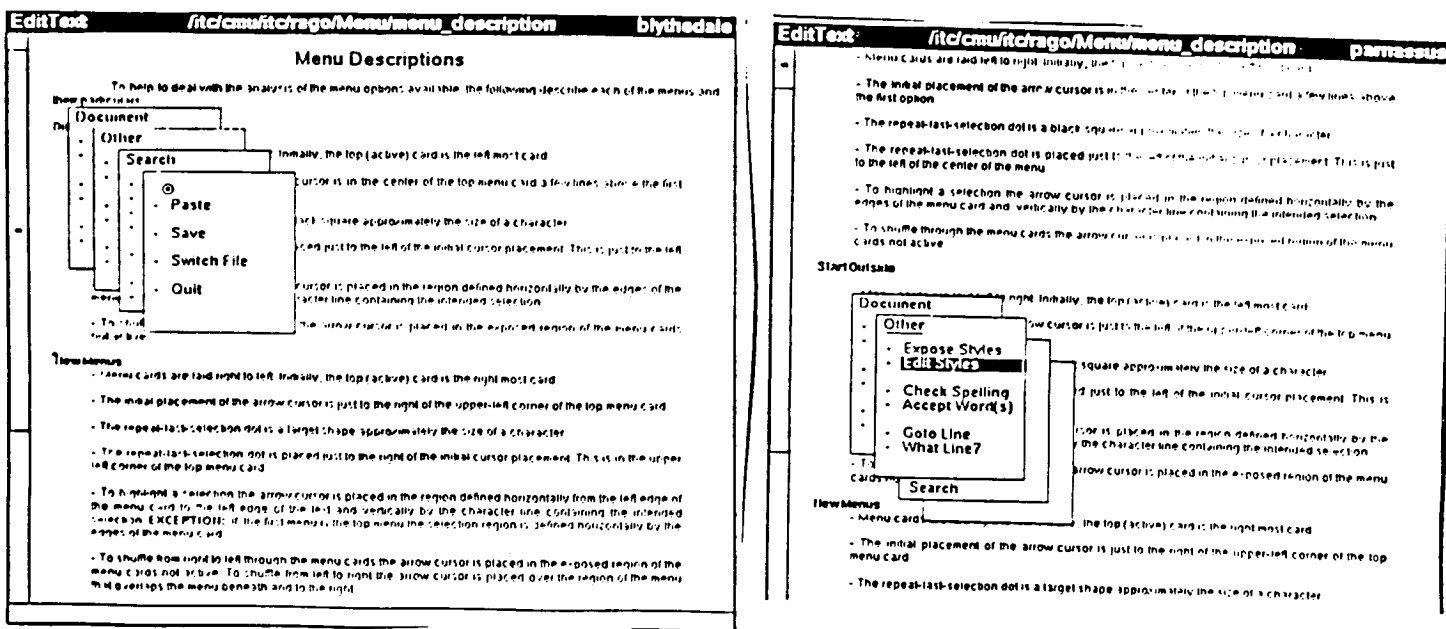


FIGURE 1b: An example of NewMenus menus

Dependent

There are two dependent variables to consider in this study: how well do novices use each menu, and how much do the novices like each menu. These indicate both the behavior and the attitudes of subjects. The dependent variables are:

- Performance: How well the subject executes the indicated tasks. Two categories of errors were observed:

- Active: Active errors are errors that result in the system taking an action. These are very stressful errors. Subjects are not always aware of making the error, nor do they know if the error is reversible. The operationalizations are:

 - Misselected* - Made the wrong selection.

 - Misrepeated* - Accidentally used repeat spot and consequently misselected an item. Thought the repeat spot was associated with the intended item.

 - Repeat and Miss* - Used repeat spot and misselected item by moving away before clicking or releasing.

- Passive: Mechanical errors in moving the arrow cursor to the correct menu selection. These errors do not manifest themselves in any way other than their own existence. They increase the time it takes to make a given selection. The operationalizations are:

 - Oscillated* - Moved off an item and back in either direction.

 - Settled In* - Oscillating more than one item away and in both directions then settled into correct item. Like a sine wave with decreasing amplitude.

 - Paged In* - Same as *Settled In* but applies to menu cards, not items.

 - Overshot* - Moved beyond any edge of the correct menu card.

 - Accidental Repeat* - Used repeat spot accidentally.

 - Repeat and Move* - Used repeat spot and moved off correct selection.

In addition to the above performance measures, the number of compound errors was recorded. Compound errors were coded as the subject making more than one type of error for a single menu selection task.

- Preferences: How much each subject preferred each menu. Affective comments made by subjects were transcribed from recordings of sessions. These were coded by positive and negative references and indicated preferences.

FINDINGS

Performance

Performance data were gathered as the subjects completed each menu selection task during the two sessions. The errors were totaled according to order and menu scheme and means were calculated from those totals.

The data indicated several interesting trends. It appears that both the **order** in which a subject learns a menu scheme and the **features** of that menu scheme have an effect on the subjects performance.

Five of the six subjects performed better with the first menu to which they were exposed. The total error rate is 18.7% higher for the second menu. An even stronger indicator of this trend is the number of compound errors made with the second menu. In five of the six cases the subjects made the same number or more compound errors in the second session. The subject was likely to make one and a half as many compound errors with the second menu scheme as the first.

There was an 18.7% higher rate of total error being made with the StartOutside menus than the NewMenus menus. In addition to the total number of errors, there were more than three times as many active errors made with StartOutside.

These findings are summarized in Table 1.

Evaluation and Comments

Affective statements made by the subjects were transcribed from the session recordings. Some interesting trends in preferences were found. Four of the subjects preferred StartOutside for a number of reasons, while two preferred NewMenus. There were no order effects to the preferences. The order subjects were exposed to the menus for which they stated a preference was evenly distributed.

The reasons for the subjects' menu preferences were focused on the features of each menu. Table 2 shows the numbers of positive and negative references subjects made for each menu scheme.

TABLE 1:

Mean Number of Errors By Order (n = 6)

Order	Mean Total	Mean Active	Mean Passive	Mean Compound
First	10.17	2.33	7.67	.833
Second	12.17	1.8	10.33	2.17

Mean Number of Errors By Menu Scheme (n = 6)

Menu	Mean Total	Mean Active	Mean Passive	Mean Compound
StartOutside	12.17	3.33	8.83	1.67
NewMenus	10.17	1.0	9.17	1.33

As the sample size is small, no significance tests were made on the data. It is only by coincidence that the mean total errors for both order and menu scheme are similar .

TABLE 2:

StartOutside

Positive
2 Shading
1 Highlighting

Negative

NewMenus

Positive
1 Navigation
1 Hash Marks

Negative
2 Navigation
1 Mouse Hole Placement
2 Hash Marks
1 Orientation

Below are some examples of the comments subjects made regarding menu features.

Positive comments about StartOutside:

Ok, one thing about this set of menus is that it uh. I like the way, on the last set I used [StartOutside]... the way it changes colors, say, as you go backwards the ones that aren't exposed are a different color. So you just, you're actually just looking. The menu that you are interested in is highlighted, where this one [NewMenus] really isn't like that.

I think I prefer this one [StartOutside] because... see how the ones in the back are darkened, it seems a little bit easier to understand which section [menu card] you are in, better than the others... the others the boxes seemed to be all jumbled together more.

I also like how the bar [reverse video highlight of menu selection] does go all the way across [StartOutside]. There's not the little things, you know, dots at the beginning of each line [NewMenus]. It seems to be a little bit more easier to like, pick out what you want.

I do prefer, you know, how it seems to put the ones you passed up already in like a background. Visually, it's like more, you can sort of imagine the pages going back.

Positive comments about NewMenus:

It seemed that the arrow, like, was going to the next one [menu card] or whichever... seemed like, you know the one [menu card] that I wanted, it went to. I didn't have to move the arrow around... as much.... when I wanted to do this. It went right over to the thing [menu card]. The arrow was already there, it seemed to me. I don't know if that was on purpose.

Both are easy to use. Maybe this [NewMenus] one a bit more, because of the dots. It's kind of like an eye key.

Negative comments about NewMenus:

Now, one thing you do have to make sure that you try to keep the cursor as close to the center of text as possible, like when you're in the menus, because if you go too close to the edge you might display the next menu.

I don't know why it doesn't seem like the items on these menus [NewMenus] stand out as much as the other ones, for some strange reason, I don't know what it is but it just doesn't 'cause it almost seems like the hash marks, aren't, they're just there, they don't look like they're really needed.

The mouse hole is inside the text [menu card]. And actually though, I don't know, I didn't try it, but. If having the mouse hole inside the text, will that automatically go back to say, if your documents or something. Like if you wanted to use your document menu....That's one thing that might confuse people. By just having it on that one thing [menu card], I think that the mouse hole is for that menu and not for all of them. Whereas having the mouse hole outside more or less it was for all of them.

Well I'm a righty and it almost seems like I have to, I'm working backwards trying to work through these menus [NewMenus]. Whereas, when they were displayed in the other direction it seemed almost easier to just slide my hand to display all, to go from menu to menu.

When asked, at the end of the testing, if they noticed anything different between the second session and the first, no subject mentioned recognizing the different menu schemes. No mention was made about the features being different until after the subject was told that the two sessions, in fact, incorporated two different menus.

ANALYSIS

There are three interesting and important trends in the data. First, there is an order dependency surrounding performance with a menu, but not preference. Second, performance was observed to be better for NewMenus, the menu scheme with more complicated mechanisms for use. Third, independent of order or performance there was a preference for the features of StartOutside.

The existence of a performance/order dependency is interesting for many reasons. The features of the menus that were used are not very different from each other. We controlled for such a dependency related to the task being performed, i.e., the menu selection tasks were the same for each subject's first and second session, but the menu they used was evenly distributed across the sessions. The differences in the menus were transparent to the subjects until they were told they existed. This dependency is important because novices learning to use a system like Andrew are likely to experience the same drop in performance when changes are made to their environment. Even small changes in the appearance of the system's user interface can be expected to effect users' performance.

Overall subjects' performance was better with NewMenus than with StartOutside, although NewMenus is more complicated than StartOutside (see Appendix A). This is inconsistent with our hypothesis that new users will perform better with menus that are less complicated to use. The mechanisms for using NewMenus are quite a bit more complicated than StartOutside and at times display inconsistencies. Even with these problems, NewMenus had a better performance record. The mechanisms that make NewMenus confusing may very well make them more accurate.

Although NewMenus had a better performance record, four of the subjects stated a preference for StartOutside. The subjects that preferred StartOutside did so because of the visual cues that it provided, mostly about shading past menus, highlighting across the whole menu card and left to right orientation. Of the two subjects who preferred NewMenus one liked the tick marks as a visual cue, the other just said, *They [NewMenus] seemed clearer, easier to understand.*

There were a number of negative comments about the features of NewMenus. The ticks making it too busy, the orientation, the navigation mechanism, and the location of the repeat spot.

FURTHER RESEARCH

To better understand the issues raised by this testing, more studies should be done in this area, more specifically, a continuation of this study with more subjects following the same procedures. Part of this continuation could include a revised version of NewMenus. One of the benefits of such a study would be to design the best menu option for new users. Another interesting question is the order dependency of performance. With a new set of subjects and different menus the findings could be validated and explanations explored.

CONCLUSIONS

Given the findings of this study and the options available, a new menu could be made as a revision of NewMenus. In this revision the visual cues that influenced the preferences of the subjects could be incorporated. This new set of menus would retain the performance of NewMenus and improve the user's perception of them.

APPENDIX A

Menu Descriptions

A menu is a set of cards that pop up in a window when the two mouse buttons are pressed simultaneously. Upon each card are a set of items, each of which is standard input to the application program running or to the Window Manager. Some of the features of all menus:

- Menu cards are displayed as a stack, with some portion of all cards visible.
- Menus have some form of a repeat spot, or mouse hole. If the arrow cursor is placed on it, the arrow will then be placed at the location of the last selected menu item.
- Menu cards may or may not have titles on them. If the menus have titles, they will be displayed in some way, at least initially.
- There is some mechanism to expose menu cards beneath the initial card.
- Menu selections are made by placing the arrow cursor in the region of the desired item and clicking or releasing the mouse buttons.

StartOutside

- Menu cards are laid left to right. Initially, the top (active) card is the left most card.
- The initial placement of the arrow cursor is just to the left of the upper-left corner of the top menu card.
- The repeat-last-selection dot is a black square approximately the size of a character.
- The repeat-last-selection dot is placed just to the left of the initial cursor placement. This is outside the top menu card.
- To highlight a selection the arrow cursor is placed in the region defined horizontally by the edges of the menu card and vertically by the character line containing the intended selection.
- To shuffle through the menu cards the arrow cursor is placed in the exposed region of the menu cards not active.

NewMenus

- Menu cards are laid right to left. Initially, the top (active) card is the right most card.
- The initial placement of the arrow cursor is just to the right of the upper-left corner of the top menu card.
- The repeat-last-selection dot is a target shape approximately the size of a character.
- The repeat-last-selection dot is placed just to the right of the initial cursor placement. This is in the upper left corner of the top menu card.
- To highlight a selection the arrow cursor is placed in the region defined horizontally from the left edge of the menu card to the left edge of the text and vertically by the character line containing the intended selection. **EXCEPTION:** If the first menu is the top menu the selection region is defined horizontally by the edges of the menu card.
- To shuffle from right to left through the menu cards the arrow cursor is placed in the exposed region of the menu cards not active. To shuffle from left to right the arrow cursor is placed over the region of the menu that overlaps the menu beneath and to the right.