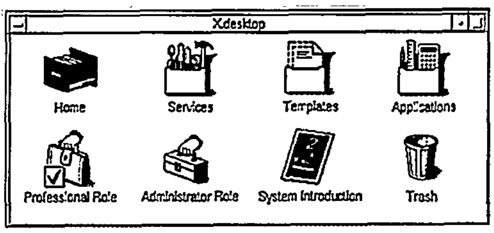


Explicitly Modal Interfaces for Business Professionals

Alan Wexelblat

photo © FPG International / Ed Braverman

W*e designed and built a desktop interface for Bull's UNIX-based Professional workstation (ProStation) targeted at business and professional users. This interface supplemented traditional desktop-style direct manipulation interaction with explicit modes and supporting views of information. The goal of this modal enhancement was to increase the facility with which the users can accomplish their daily tasks while using the system. We expected the ProStation users to be generally unfamiliar with workstation-style computers, though possibly having had minimal exposure to PCs. • Before starting design of the interface, we conducted a series of observations and interviews with business professionals. Insights from this process led us to design and create a moded interaction style and influenced the elements chosen as parts of the interface.*

Target Users

The intended users for the ProStation were business professionals, sometimes referred to as "knowledge workers." In general, these are people who are expert in their domain of operation—such as business, finance, insurance, or personnel—and who primarily work with information (or knowledge) rather than with material objects.

Despite their domain expertise, these people were expected to be initially unfamiliar with computers. Their computer knowledge was usually limited to the use of one or two applications on a Mac or PC. Our interface needed to hide these complexities while giving users the power such a workstation could offer.

We targeted the subset of these knowledge workers who had, by and large, outgrown their personal computers. Simply giving them a faster CPU or more memory for their PC would not have helped them get their jobs done any better. We wished instead to give them a different computer work environment, one with multitasking, windows, a large display and a more powerful computer engine. This introduced significant additional complexity at the interface and we were required to cope with this as well. All the while, we had to bear in mind that even though these people did not produce physical artifacts as part of their daily work, the computer to them was simply another tool. The interface had to enhance their feeling of using a tool and had to emphasize the similarities between their paper-based and computer-based work environments.

Initial Study of Users

Despite having the idealized description of our intended users laid out above, we did not—at the start of the interface design process—know enough about the types of tasks our users wanted to accomplish. We viewed the computer as a cognitive prosthesis; that is, something which would aid these users in manipulating their information. To achieve this, we needed some idea of what our users would actually do in their daily work. Unfortunately, since the ProStation was a new product, we had no installed base of users whom we could ask.

So we worked with business professionals at Bull who fit the intended user profile and who were likely to be early users of the workstation when it was delivered. During design, we expected the workstation to be made generally available, though in the end it became only an internal product.

As with many early user-centered design efforts, time for user observation had to found during the development/coding activities. Management was distantly supportive of these human-factors efforts, but did not agree to adjust the delivery schedule to account for the extra time. Given these constraints, we were able to identify and observe ten knowledge workers in the time we had.

These professionals held a wide variety of jobs, from a Production Assistant in the Publications department to a Contracts Manager for Desktop Products. All these persons had a computer (Mac or PC) in their office; the computer was available to them whenever they needed it. The software on the computers ranged widely as well. In general, the subjects had, or could have had, any computer program they wanted to support their jobs.

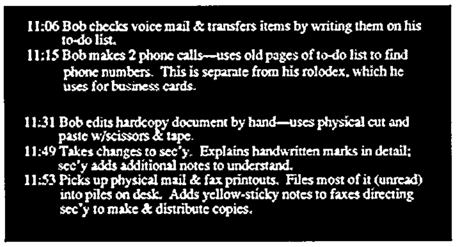
All the subjects were college-educated and had at least two years' exposure to computers. The level of average daily computer use varied from 5-6 hours per day to less than twenty minutes. All had been in their current jobs (or an equivalent job at another company) for at least a year.

We observed each subject by following him or her around during the course of a normal day. "Normal" is, of course, a matter of debate; inevitably, in some ways, things happened in these subjects' lives which made their days unique. When we scheduled the observations, we asked the subjects to select days which they expected to be more or less typical. In one or two cases, subjects remarked during observation that the day was exceptional in some specific way, but the majority

The interface had to enhance their feeling of using a tool and had to emphasize the similarities between agreed that the days we had observed were considered normal.

The goal of these observations was to determine the sorts of tasks performed by the business professionals which we might support. To this end, we recorded the activities of each subject. A sample from these activity logs is reproduced in Figure 1.

We did not explicitly code the subjects' activities according to any pre-determined scheme; rather, we highlighted actions and noted how subjects manipulated their job-related information. For example, we noted in the log shown in Figure 1 that Bob used previous days' To-do lists to retrieve the phone numbers he needed to make today's phone calls.



11:06 Bob checks voice mail & transfers items by writing them on his to-do list.
11:15 Bob makes 2 phone calls—uses old pages of to-do list to find phone numbers. This is separate from his rolodex, which he uses for business cards.

11:31 Bob edits hardcopy document by hand—uses physical cut and paste w/scissors & tape.
11:49 Takes changes to sec'y. Explains handwritten marks in detail; sec'y adds additional notes to understand.
11:53 Picks up physical mail & fax printouts. Files most of it (unread) into piles on desk. Adds yellow-sticky notes to faxes directing sec'y to make & distribute copies.

Figure 1

Two excerpts from Activity Log

Observations lasted five to six hours per subject. A few days after the observations, a follow-up visit was made. This second visit was structured around a series of questions designed to illuminate the subject's experiences with currently-available computer systems and their impressions of their work environments. The questions were open-ended and served as guides for subjects to discuss their perceptions of their work environment.

Two questions were most enlightening from an interface-design standpoint. We asked subjects what the "objects in their world" were. People gave us laundry lists as expected, often looking around their offices as they listed off things they saw; the surprising thing to us was that the lists were very similar and not long. This is explained in more detail in the next section.

The other enlightening question was "Please describe your filing system" — a question intended to elicit information on how subjects arranged their knowledge-bearing objects. Subjects' descriptions varied wildly, including alphabetical, chronological, and topical systems. Surprisingly, all subjects indicated that their filing system was evolved, rather than predetermined. No one had a set of categories they started with and continued to use. Many factors contributed to this evolution, but the unanimity of answers convinced us to abandon embryonic notions of pre-structuring the user's filing environment on the ProStation.

Important Positive Findings

Given that we only observed one worker in each of ten areas in the company, we did not see any common detailed patterns of activity. Nor were we able to be very specific about the tasks that would be performed by our eventual system users. In an ideal environment, we would have been able to observe more people and draw statistically supportable generalizations. As it happened, we ended up depending on insights based on our experience of living with the subjects for a day.

During the second (question-and-answer) observation session, reactions to our suggestions for automation of various office objects/tasks produced decidedly mixed results. Some subjects were quite receptive; these were the people who were most swamped by their current tasks. Others had worked out systems using non-computer tools and were highly resistant to changing what worked. One subject's remark was typical:

"I know other people have more efficient ways of doing this. I just don't think their way would be better for me. This is something that I know works."

We were able to glean several important insights from our user interactions. These insights shaped the Professional Environment interface we then developed:

Professionals deal with their computers in two distinct ways. Most of the time they view the computer as a tool—like paper and pen-

their paper-based and computer-based work environments.

cil—which they use to complete their tasks. Their tasks are describable in domain-dependent and computer-independent terms: "writing a letter," "reviewing a budget," etc.

From time to time, though, these professionals perform tasks in which the computer is an end rather than a means. These tasks are described in computer-specific terms: "backing up the system," "installing new software," etc.

Professionals have a distinct, limited set of objects types used in their normal tasks. These types describe familiar things, some of which are special to the business professional's domain. These things were drawn from the list of objects given by the subjects (as noted above). They are: Action List, Budget, Chart, Contract, Document, Expense Report, Mail Message, FYI Note, Letter, Memo, Note, Phone Message, Plan, Proposal, General Report, Market Report, Routing List, Schedule, To List, To-do List.

From time to time, though, these professionals perform tasks in which the computer is an end rather than a means.

Professionals claim to value ease-of-use over ease-of-learning. However, we noticed that templates seemed to be critical predictors of the level of system use. Several times we observed professionals not using available software because they were unwilling to invest the time to create a form in addition to the time needed to do the task at hand. This forms creation process appeared to be seen by them as different from the software-learning time.

Observations seemed to indicate that what was desired was something closer to ease-of-affordance (in Norman's sense [Nor88]) than what we typically think of as ease-of-use—that is, they wanted an obvious easy entrypoint into the system or the tool, even if the program was complex to use.

These findings are embodied in the interface architecture, as described in the next section.

Interface Architecture

Our interface architecture was built to support the insights described above. The following sections show how features of the interface relate to and support these insights.

Roles

To support the two distinct ways of seeing the computer described in the first insight, we created two explicit modes. These modes do not change the way the user interacts with the system, as in vi's edit and insert modes. Instead, the modes provide different views of the user's information, and provide different sets of tools. The modes are conceived of as filtering and organizing mechanisms, similar to Xerox PARC's Rooms mechanism where different spaces are created within the computer environment and each space is populated with a distinct set of capabilities. We called our filtering/organizing mechanism a "role."

The concept of role has been explored in the organizational literature (for example, by Lucy Suchman or Ronald Lee). Roles are used to divide up and help understand the tasks performed by individuals in an organization. While this to some extent describes the engineer's view of roles, we wanted our users to think of the roles in the system as being more like roles in a play, as described by Brenda Laurel in Computers as Theater. In this conception, a role provides possibilities for action, guidelines, and useful constraints. It suggests ways of thinking about the interaction. We intended users to view the roles as being like suits of clothes they wear.

Our on-line System Introduction and paper documentation explained roles by asking the user to consider first what kind(s) of tasks she would be doing. Users performing everyday tasks were led to select the Professional role, the default mode in which the system starts up. Users who wanted to do administrative tasks were led to switch to the Administrator role.

In order to make the interface mode—which role is currently active—more apparent, icons

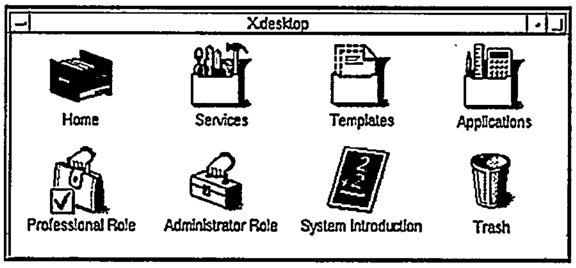


Figure 2
Standard Desktop

representing both roles were placed on the user's desktop, as shown in Figure 2. The currently-active role is indicated by the check mark. Users could switch roles by making a menu selection or by double-clicking on a role icon.

Format and Purposes

As noted above, our users had a limited, domain-specific, vocabulary for their task objects. They described things as "letters," "memos," "budgets," and so on. However, this way of referencing things is not well supported by most commercial systems. Systems such as the Macintosh Finder which specify, or allow the user to specify, types for files do so by representing the storage format of the files. For example, Microsoft Word data files all resemble the Word executable.

This information is valuable for engineering purposes, as it allows the underlying system to take appropriate action when, for instance, the user wishes to edit a file. We preserved and used this extensional information in the Professional Environment. However, the business professional does not often care if a file is in Word or FrameMaker format. Instead, she would rather know which file is the letter and which is the memo. Most systems require the user to incorporate this information in other characteristics of the file, such as the name.

Our observations of the subjects showed that they often recognized items by their appearance (such as thickness) or position (bottom of the stack). For example, one subject on the day we observed him needed to assemble all comments before revising a memo. He spent several minutes looking for "the fax from Europe." When asked how he knew it was missing, he replied:

"...it's obviously not there. I can see all the edges, and I know the Europeans use [longer pages]. Since there's nothing sticking out of the pile, I know it's somewhere else..."

To support this kind of knowledge we created a second "type system" for files in the Professional Environment, one which records the intentional information about the users' files. We called this the "purpose" of the file because it recorded the reason why a file was created and for what purpose it was used.

Specifically, we took the list of user object types given during the interviews (Action List through To-Do List) and created mechanisms whereby users could associate one of these types with each file they created. This file purpose was represented in the icons shown on the desktop when the user was in the Professional role.

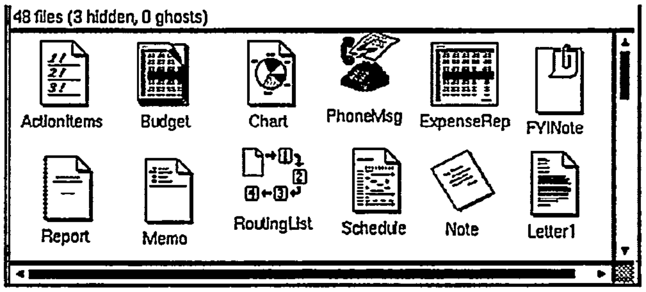


Figure 3
Purpose Icons of Documents

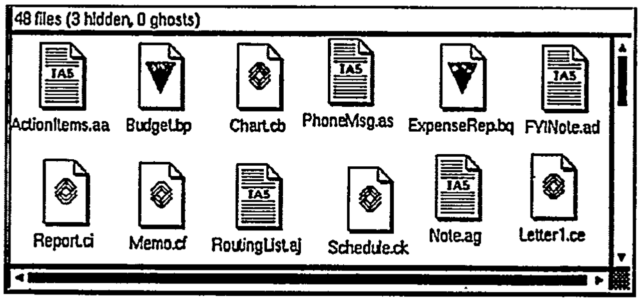


Figure 4
Format Icons of Documents

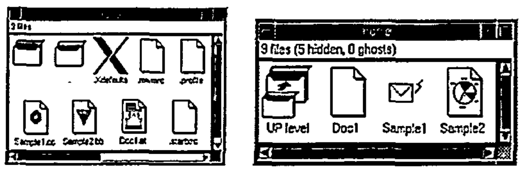


Figure 5 **Administrator and Professional views of a Home Directory**

Iconic Representations

As shown in Figure 3, the icons were drawn to visually represent the purpose of the files. Memos look like memos, action-item lists look like lists of action items, and so on. This enabled users to pick out quickly the file or files they wanted and presented the business professional with an environment composed of familiar objects. We expected our users to operate in the Professional role—and thus, to use this representation—ninety percent of the time.

If users wanted to view files by their format, they needed only to switch to the Administrator role. When users did this, the system redisplayed all the desktop objects. Figure 4 shows the same set of files as Figure 3, but viewed while in the Administrator role. This allowed users to see which program created which files, as the format icons use symbols from the applications.

In addition to changing the way users saw their information, the roles filtered out extraneous detail. In the Professional Environment, it was assumed that a user acting in the Administrator role had more UNIX knowledge than a Professional. Therefore, the Administrator view showed all files in a directory, even the "dot" files such as .login, .Xdefaults and so on. Figure 5 shows a user's home directory viewed in the Professional and Administrator roles.

The Administrator role also made available some functionality—in the form of system services—which assumed the user had some UNIX knowledge (for example, a UNIX command-line xterm and UNIX man pages). Figures 6 and 7 show the system Services container in the Professional and Administrator roles.

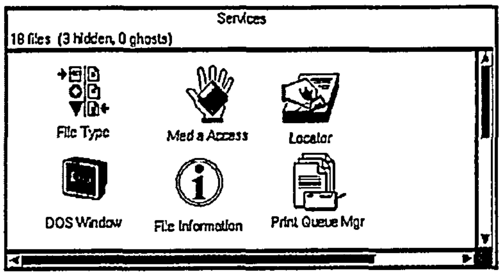


Figure 6 **Professional Services**

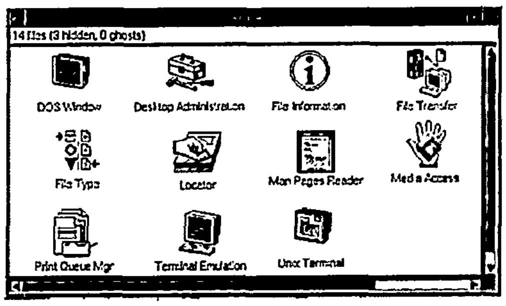


Figure 7 **Administrator Services**

Interaction Style

Regardless of which role the user operated in, the interaction style remained the same. We felt that what we call an "object style" interface would best support our users' expectation of ease-of-affordance. We felt it was important that every object, when "poked" by the user, would respond in some way. Everything visible was an entrypoint into some software action.

To do this, we used pop-up menus extensively to supplement the usual across-the-top menu bars. Underlying code interpreted menu selections as messages sent to the objects, with the class of the object determined by its storage format, and invoked appropriate methods. For example, selecting "Edit" from a document menu sent an "edit yourself" message to the document. On a FrameMaker document, for example, this resulted in the document being opened in a new Maker window.

Menus were also context sensitive. This avoided overwhelming the user by presenting only operations relevant to the current context. Two examples of this are shown. In Figure 8, the top left-hand side shows a template and its pop-up menu. The right is a standard document pop-up menu, seen once the template has been instantiated. The bottom shows the pop-up menu for either document when it has been put in the trash. Nine possible document actions are divided into three contexts—all and only the actions for that context are shown.

This subdivision applied to system objects as well as user objects. In Figure 9, the left-hand side shows the desktop wastecan in its empty state. The right-hand side shows it as it appeared with items in it. Not only did the appearance change, but the menu offered additional actions which were appropriate for its new state.

Templates

As noted in the section on insights, templates were deemed to be extremely important.

Therefore we made templates a "first class" object and placed them in their own container on the user's desktop. Our object-style interaction allowed templates from any application to be placed in this one container. All templates responded to the same "duplicate yourself into" message, which allowed users to make their own copies of templates. Users could then edit the template as they would edit any other document.

Base Technology

Our chosen base technology was IXI's X.Desktop product, a graphical interface to the UNIX file system. X.Desktop provides the ability to display files and directories as icons, and to manipulate these icons directly with the mouse. Directory icons can be opened to display their contents as other sets of icons in a new window.

X.Desktop also provides a special "desktop window," similar to the root window in X or the background window in Finder or Microsoft WIndows. In this special window, iconic actions may be taken, which do not correspond to a specific file-system directory. In the Professional Environment, users could save and switch among configurations of this desktop window. This allowed users to have multiple configurations of their computer environments and to evolve a comfortable and efficient electronic environment just as they did with their paper-laden desktops.

Our interface is built as an enhanced layer on top of the basic product, written using IXI's rule language, shell scripts, and C programs. The interface runs as a seamless whole; the user does not know which part is implementing which action. Nevertheless, the interface can still be customized to a significant degree. The enhanced interface, together with supporting integrated programs (such as FrameMaker) was shown to users simply as the Bull Professional Environment.

Evaluation

In an ideal environment, we would have returned to our original interviewees and observed their behavior and with the new machines. Unfortunately, in our highly-mobile product world this was not possible—more than half were no longer with Bull when the workstations were distributed within the company. Management also expressed a desire to understand how the product might be viewed by non-Bull users.

Therefore, we conducted a series of user tests of the first version. Nine subjects in four sessions tried to do a series of tasks in which they were presented with a folder of documents and asked to create subfolders and sort the documents into the subfolders. Once the organization task was done, their instructions were to edit a document, but none of the test users completed the original task in the allotted time. Eight of the subjects worked in pairs using a codiscovery paradigm where they were encouraged to talk to each other. One subject worked alone and was encouraged to think aloud by the tester. Test sessions lasted two hours each.

Codiscovery is a term for what happens when people working together share a joint insight. One person will think, for example, "Let's try moving the document." Since both people are peers with shared access to the mouse/keyboard, the person with the idea will more often explain what she is doing. If she does not, her partner frequently asks "Why did you do that?" The resulting explanation leads to a shared discovery of the results. By videotaping this interaction, the development team can learn test users' thought processes as they work with the system.

The subjects were recruited based on their experience with DOS PCs and lack of experience with desktop-type interfaces. The subjects

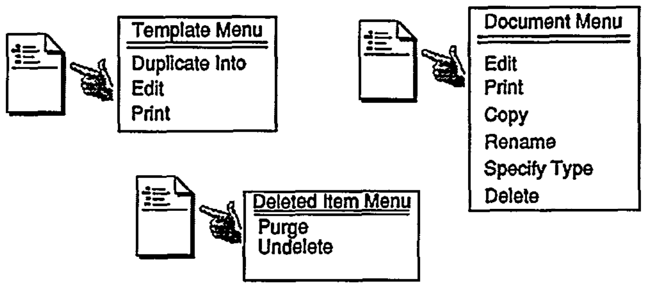


Figure 8
Content-Sensitive menus for a file.

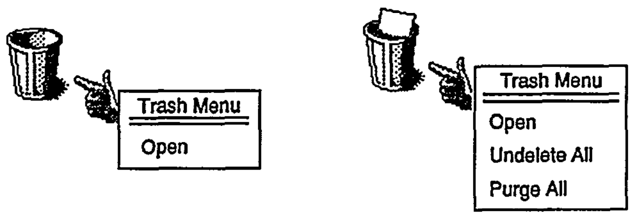


Figure 9
Content-Sensitive wastecan menus.

worked without prior knowledge of the Professional Environment and without reading documentation. They were given a twenty-minute training session in the use of the mouse and how to manipulate windows.

Results

Testing revealed some problems with the Professional Environment interface, particularly related to subjects' inability to manipulate the mouse. There were also several positive findings relevant to the special extensions we made to the standard desktop model.

The visual typing afforded by the document purposes was understood intuitively by all subjects. One subject remarked, when asked why he had moved a document to a particular subfolder:

"[The directions say] say to put reports in this folder. This looks like a report; it has these things on the side [subject gestures at the part of the icon designed to look like a spiral binding]"

Roles were also well-understood, even though the test task did not involve use of the roles explicitly. One pair of subjects had the following exchange:

- Subject 1: "Let's try this administrator role thing."
- Subject 2: "Nah, we're looking for help. Administrators usually know what they're doing already."

The test task did not involve the use of templates; however, one subject pair—while trying to figure out how to create a subfolder—opened the Templates container looking for a "down level" template. Their conversation indicated that they understood that templates were how one "made a new thing" and they were indeed trying to make a new thing. We added a Down Level template to the next version of the interface. This template icon was a wrapper around a script which created a subdirectory in the location to which it was copied.

The one area in which the interface consistently failed was the pop-up menus. Users had a hard time finding the background popup menus, even though the initial instruction demonstrated their use. Based on watching the test subjects, we confirmed that the background does not provide the correct affordances. Users did not know there was anything there they could "poke" to get a response. To fix this problem, we enhanced the menu bars in the next version of the Professional Environment. The revised menu bars completely duplicated the functionality of the background popups, but provided constantly-visible items on which users could click to get ideas for things to do.

In an ideal environment, we would have been able to repeat these tests with the revised interface and seen whether or not they solved the problems we had discovered. Unfortunately, as noted before, the schedule did not have time to account for testing/revision cycles and we were required to ship the revised interface as the product interface and move on to other tasks.

Conclusions

We believe we created an unique interface. Using nothing technologically revolutionary, we were able to enhance significantly a traditional (desktop metaphor) interface for a computer-naive population, despite hosting that interface on the traditionally user-unfriendly UNIX environment. This represents an important incremental improvement in desktop interfaces. Our work with roles and file purposes suggests that it is possible to capture more of the user's cognitive model in the interface, provided that the domain of work is well-circumscribed. We were able to do this despite an inability to predict in detail in advance the tasks our users would be attempting.

Acknowledgements: I would like to thank my colleagues at Bull who reviewed the ProStation interface design and participated in its implementation. In particular, Mary Ann Wolf helped in the user interviews and observations. Jared Spool of User Interface Engineering conducted the human-factors evaluation.

Author's present address: MIT Media Lab, 20 Ames St. Cambridge MA 02139 wex@media.mit.edu. The work described in this paper was done in 1990-1991 when the author was the primary user interface designer/implementer for the Professional Workstation product.

References

[1] Henderson, D.A. & Card, S. K. "Rooms: The Use of Multiple Virtual Workspaces to Reduce Space Contention in a Window-based Graphical User Interface," *ACM Transactions on Graphics*, 5(3), July 1986.

[2] Laurel, Brenda, *Computers as Theater*, Addison-Wesley, New York, 1991.

[3] Lee, Ronald M. "Bureaucracies as Deontic Systems," *ACM Transactions of Office Information Systems*, 6 (2), April, 1988.

[4] Norman, Don. *The Psychology of Everyday Things*, Basic Books, New York, 1988.

[5] Suchman, Lucy. *Plans and Situated Actions*, Cambridge University Press, Cambridge, 1987.