Browsing Unix Directories With Dynamic Queries: An Evaluation of Three Information Display Techniques

Holmes S. Liao, U. of Maryland.
Masakazu Osada, Toshiba Corporation
Ben Shneiderman, Dept. of Computer Science, U. of Maryland

Human-Computer Interaction Laboratory
& Deparment of Computer Sciences
University of Maryland
College Park, MD 20742-3255
(301) 405-2725
e-mail: ben@cs.umd.edu

Abstract:

We designed, implemented, and evaluated an innovative concept for dynamic queries which involves the direct manipulation of small databases. Our domain was directories in a Unix file system. Dynamic queries allow users to formulate queries and explore the databases with graphical widgets, such as sliders and buttons, without requiring them to have any knowledge about the underlying structure of the database query languages, or command language syntax. Three interfaces for presenting directories were developed and tested with eighteen subjects in a within-subject design. The results of the formative evaluation yielded some useful guidelines for software designers.

1. Introduction

The conventional approach to database queries is to compose a textual query in a language such as SQL, which presumes that the users are familiar with the logical structure of the database and the syntax of the query language.

Dynamic query is an approach which combines both direct manipulation [1] and database visualization [2]. The database is represented on the screen in graphical form, thus providing continuous visual feedback. As the user manipulates graphical widgets, such as sliders or buttons, a sequence of queries are "fired" by the dynamic query program, and the database is searched based upon the settings of the sliders. In a dynamic query program, rapid, incremental, and reversible control of the sliders plus instant visual response to the queries is essential.

Dynamic queries have been shown to be highly advantageous in finding information about the chemical table of elements [3] and in a real estate database [4]. With dynamic queries, users who want to retrieve (somewhat confined) information from a database are spared the effort of specifying SQL commands. The SQL trial-and-error specification process is replaced with the use of sliders. Each slider in a dynamic query program represents a distinct database query variable [3]. As the user moves the slider (i.e., a query variable changes), the query and its results are updated and displayed on the computer screen immediately. This approach fits well in cases where the value of selection criterion are continuous and quantifiable, and the entire database can be displayed and viewed on the screen in a convenient way.

For example, when users find that a hard disk has almost no space left, they might consider deleting some of the oldest files with larger-than-average file sizes in order to recover some disk space. Since a file listing is effectively a database, determining which files are the best candidates for deletion requires considerable effort in searching throughout the disk using commands or utilities with conventional visual inspection of lengthy lists. If the user employs a dynamic query directory browser in which every attribute of a file becomes a query variable which can be manipulated dynamically, the complexity of the task can be reduced dramatically.

There are limitations to dynamic query. Since the query requires rapid response time, an efficient data structure and search algorithm is required. A powerful graphics workstation is usually needed to display the output of dynamic queries. Other challenges for designers of dynamic queries include: finding a suitable visual display for output, managing large numbers of sliders and buttons, and providing complete boolean queries (AND, OR, and NOT) [8].

In this paper, we describe a formative evaluation of three different user interfaces for a dynamic query directory browser. Most directories are organized with several levels in a tree-like hierarchical structure [5], but this experiment does not include the navigation features of a typical Unix directory browser. Rather, the directory browser employed in this evaluation focuses on finding files using their attributes, such as modification time and file size. Section 2 contains the description of the three interfaces, hypothesis, experimental variables, and the procedure. The result of the statistical analysis (t-test) is presented in section 3. Section 4 includes the interpretation and the discussion of the statistical results. Conclusions and future directions are given in section 5.

2. Formative Evaluation

2.1 Introduction

This formative evaluation compared three different interfaces for database query and visualization. The first version, ASTER (see Appendix A-1), displayed every Unix "ls" entry on the screen, and showed an asterisk (*) in front of each entry line that matched the selection criteria. The second version, COLOR (see Appendix A-2), was similar to the first one but used different colors (blue for selected, yellow for de-selected) instead of asterisks to identify the entries selected by the current query. The third version, EXPAND (see Appendix A-3), displayed only those entries that matched the dynamic query. Since the length of the matched file listing varied, this interface had an expanding and contracting visual effect; therefore we named it EXPAND. After the implementation was completed, we discovered that the impact of expanding/contracting was so dramatic that it had a rather unpleasing "jumping" visual impact. We then modified the program to delay the update of the file listing until the user released the mouse button. During the pilot study, we found out that the second version of EXPAND interface provided a more comfortable display effect than the first one. For the directory browser evaluation we tested only two selection criteria, file size and date last modified.

We defined the size of the directories relative to the size of a screen. In our experimental programs, each screen could display up to 32 lines of text. Two sizes were tested, small and medium. A small directory contained less than one screen of file listings (around 30). A medium directory contained more than one screenful but less than two screens of listings (around 60). We chose 60 lines of text as the medium size because we wanted to add a scrolling factor into our evaluation. Users scrolled through at most two screens when working with the medium directory.

2.2. Description of Interfaces

All three interfaces were built using Sun Developer's Guide user interface prototyping package in the OpenWindows 2.0 environment. Some Xview and Xlib graphics primitives were used as well. The prototyping and development were implemented on the Sun SparcStation 1+ workstation with a 17 inch color monitor. Some Sun IPCs were also used to conduct this experiment. (The Sun IPC has the same CPU as the SparcStation 1+. Clock speeds on both machines are the same.) Each of these workstations was equipped with Sun's standard optical three-button mouse. During the evaluation, the mice were the users' only input device.

A 14-point Courier font was used across all interfaces for the text in file listings. In the COLOR interface, visible texts were dark blue on a white background, and invisible texts were light yellow on a white background. For the other two interfaces, ASTER and EXPAND, dark blue was the only color used for text display on a white background.

Scrolling methods were the same for all three interfaces. Users could scroll forward or backward, one line at a time, or one screen at a time. They could also go directly to the end or beginning of the "ls" file listing. The users could go to any part of a file listing by sliding an elevator in the scroll bar to a relative point on the scroll bar. The entire height of the scroll bar represents the entire length of the file listing.

There were two sliders provided to each interface, an age slider and a size slider. The age slider was used to specify the age of the file by the number of days this file was last modified starting with the current date and counting backwards. The size slider was used to specify the file size in Kbytes. There was a pair of activate/de-activate radio buttons with each slider to activate and deactivate the criteria for a query. Using the combinations of the activate/de-activate radio buttons, a user could specify that either size selection criteria was effective, age criteria was effective, both were effective, or neither was effective. There was a second pair of "greater than" and "less than" radio buttons with each slider to specify a query variable condition. For example, to give a query like, "list all the files that are greater than 7 Kbytes," a user had to deactivate the age slider, activate the size slider, push the ">" radio button associated with the size slider, then move the size slider's drag box to 7K position.

2.3 Experimental Variables

Independent Variables:

I. Type of interface format:
   - i. ASTER
   - ii. COLOR
   - iii. EXPAND

II. Size of the directory:
   - i. "medium", roughly 60 files
   - ii. "small", roughly 30 files

Orders of the three interfaces during the experiment were counter-balanced. Participants were assigned to follow these combinations in a impartial way to avoid any bias caused by the users' learning effect. Training sessions were designed with the same principle.

Dependent Variables:

- i. Time to find an answer to each question
- ii. Subjective satisfaction

2.4 Participants

Eighteen subjects participated in the experiment. The prerequisite for participating in this experiment was knowledge of basic concepts about computer files. As it turned out 16 of our 18 participants were expert computer users and had experience with at least 3 kinds of computers. The dominant interaction feature of our experiment, the mouse, had been used by 15 of the 18 participants. Eleven participants were between the age of 20 and 30 years old. Five participants were between 31 and 40 years old. Two were forty years old. Six of the eighteen participants were female.

2.5 Materials

To eliminate users' tendencies to memorize a directory listing or layout, we used different but closely-resembled directories for each interface. The six directories used for the experiment all contained text information, such as e-mail correspondence or technical documents. The ten timed tasks used for all interfaces are listed in Appendix B. We found the standard Questionnaire for User Interface Satisfaction (QUIS) [6] was too long; our version contained 15 of the original 72 questions plus four of our own which assessed the use of slider and compared the interfaces with other directory browsers (or browsing commands). Our modified version of QUIS appears in Appendix C.

2.6 Procedures

Each evaluation session lasted approximately ninety minutes. All participants worked on either Sun SparcStation 1+ or Sun IPC workstation. The tasks were always completed first for the small directory and then the same set of tasks were done for the medium directory. Each session consisted of the following phases:

1. Introduction and Training: Participants were given an introduction to read that described the purposes and procedures of the evaluation. They were then asked to sign a consent form. Then

the appropriate training document was used as a guide to explain the mouse and scrolling features as well as any specific features of the interface they were using. The document also explained how to scroll the text continuously, line-at-a-time, page-at-a-time, and jump to any part of the file listing.

2. Practice tasks: To be sure the participant understood all the required functions of the interface, ten practice tasks were given that were similar to the real timed tasks they would do. During this phase the tasks were not timed and the experimenter could explain anything the participant did not understand.

3. Timed tasks: The participants were required to answer ten questions (appendix B) about information in the current directory. Each question was read and understood, and then the participant started work towards answering the question. After each written response, the two sliders were reset by experimenters. These tasks primarily involved finding information and the participant was not required to find the correct response. If an incorrect response was made they were not informed that the response was incorrect but would proceed to the next question. The time taken by each participant to answer correctly or incorrectly each of the ten tasks was recorded. A digital handheld stopwatch was used.

4. Subjective Evaluation: The participants were given a subjective questionnaire after they had completed all of the tasks for one interface. The questionnaire asked about the participant's opinions on the interface features and their relationship to completing the tasks.

2.6.1 Administration

The participants were advised to act as if they were actually looking for information, without feeling rushed or pressured, but not to waste any time either. The experimenter sat next to the participant, administered the tasks, reset the sliders' values to 0 after each response, and followed the proper timing procedure. This experiment was conducted over a period of 2 weeks.

2.6.2 Timing

Studies have shown that the human factors involved in online search strategy include the cognitive abilities of the participant when selecting methods for browsing and the reading speed at which users read or scan the material [7]. In an attempt to minimize the difference in reading and comprehension speed, the timing of each browsing task was begun once the participants fully understood the question. While reading the questions, the participants were asked to face away from the screen. After the participants signaled that they were ready to begin working toward an answer, they were asked to face the screen and as soon as they clicked on the first mouse button the timing began. This method reduced the differences in reading and comprehension rates among participants and lessened the variability of completion times and disorientation some participants might have suffered if the screen was left at the completion position of the previous task.

2.6.3 Grading

During the timed tasks, there would sometimes be uncertainties as to whether the participant correctly completed a task. In these cases, the experimenter would stop the stopwatch and ask the participant questions about how the task was completed or why a particular answer was given. If the participant responded with enough knowledge of the correct answer, then the experimenter would accept the participants' (poorly worded or incomplete original) answer and record the elapsed time. If the discussion between the experimenter and participant did not convince the experimenter that the participant really knew what the correct answer was, the experimenter would mark down and time and that the question was answered incorrectly and the participant would be asked to move on to the next question.

2.7 Hypotheses

Our main hypothesis was that the EXPAND interface, which displayed only the entries that satisfied the query, would be rated the highest in user satisfaction and would facilitate faster and more accurate performance, especially when used in the medium directory. Because of their similar display methods, we hypothesized that ASTER and COLOR would not have significant performance differences when used in both small and medium directories. COLOR, however, would be rated higher than ASTER in user satisfaction because of its visual effects and its effectiveness in identifying the entries that satisfied the current query.

3. Results

For timed tasks, all statistical analysis for the three interfaces were done for both small and medium directories using the t-test method for each task separately. QUIS data wa analyzed separately using the same method. Mean time and standard deviation of each timed task for each interface in both small and medium directories are shown in table 1.
Results of the t-test are shown in table 2.

	ASTER(S) Mean(SD)	ASTER(M) Mean(SD)	COLOR(S) Mean(SD)	COLOR(M) Mean(SD)	EXPAND(S) Mean(SD)	EXPAND(M) AVG (SD)
Q1	25.3(22.8)	55.9(30.4)	27.2(14.7)	34.7(17.6)	25.5(9.64)	23.8(11.5)
Q2	45.1(26.6)	45.8(24.9)	35.4(16.5)	66.7(23.9)	27.4(13.4)	28.1(11.9)
Q3	34.3(17.9)	55.7(36.6)	33.8(16.8)	43.0(17.0)	26.8(8.22)	24.8(8.34)
Q4	41.6(25.4)	49.4(21.0)	43.8(17.2)	37.5(14.9)	40.8(12.1)	21.5(13.2)
Q5	32.2(15.2)	41.4(17.6)	39.4(23.0)	36.7(14.3)	32.1(23.5)	27.7(14.4)
Q6	30.3(13.3)	45.4(18.5)	29.6(9.77)	37.1(13.2)	24.1(7.20)	22.3(9.97)
Q7	46.8(26.4)	88.9(25.8)	56.7(37.4)	70.2(25.9)	47.5(22.3)	77.2(40.9)
Q8	43.9(30.1)	59.5(46.5)	33.8(21.4)	50.4(36.0)	45.3(38.1)	45.2(24.1)
Q9	42.7(33.4)	50.4(43.0)	31.8(16.9)	45.8(24.9)	38.7(23.2)	41.2(32.9)
Q10	37.3(24.0)	27.7(19.7)	38.7(22.7)	34.1(23.7)	32.1(18.7)	27.5(22.2)
Total	380 (164)	520 (204)	370 (135)	456 (125)	340 (93.8)	339 (121)

Table 1. Mean Time and Standard Deviation for Timed Tasks

Table 1 shows that EXPAND had the smallest mean time for small directory among the three interfaces. As the results of the t-test show, there was almost no statistically significant differences between the three interfaces for small directory (Table 2). Because of the large number of t-ratios, we set the significance level at 0.01. In Table 2, the positive numbers indicate that the bottom interface was superior to the top one.

	ASTER(S) vs. COLOR(S)	ASTER(M) vs. COLOR(M)	ASTER(S) vs. EXPAND(S)	ASTER(M) vs. EXPAND(M)	COLOR(S) vs. EXPAND(S)	COLOR(M)vs. EXPAND(M)
Q1	-0.3	2.56	0	4.18**	0.42	2.2
Q2	1.31	-2.6**	2.51	2.72**	1.58	6.15**
Q3	0.09	1.33	1.63	3.49**	1.6	4.08**
Q4	-0.3	1.97	0.11	4.78**	0.6	3.42**
Q5	-1.1	0.87	0.03	2.55	0.95	1.89
Q6	0.17	1.57	1.73	4.67**	1.92	3.78**
Q7	-0.9	2.17	-0.1	1.02	0.89	-0.6
Q8	1.16	0.65	-0.1	1.16	-1.1	0.52
Q9	1.23	0.39	0.42	0.72	-1	0.47
Q10	-0.2	-0.9	0.73	0.03	0.95	0.86
Total	0.19	1.13	0.88	3.23**	0.77	2.86**

**: significant difference at $p < 0.01$ level

Table 2. t-ratio for Timed Tasks (positive values favor the bottom interface)

For the medium directory, EXPAND interface also had the smallest average mean time for all tasks combined. The t-test results show statistically significant differences for the total time between ASTER and EXPAND and between COLOR and EXPAND while there was no significant differences between ASTER and COLOR. The EXPAND interface had advantages for several tasks for the medium directory. Although there was one task which showed significant difference between ASTER and COLOR, ASTER had some timed tasks that exhibited shorter mean time while COLOR had other tasks that exhibited shorter mean time. Consequently it cannot be inferred from these results that the COLOR interface was more efficient than the ASTER interface.

For timed tasks Q8, Q9, and Q10, where participants were asked to estimate average size, average age, and a majority range of size, respectively, there were no significant differences between any two interfaces, either in small or in medium directory.

We calculated the error rates based on our grading scheme (Section 2.6.3) for ASTER, COLOR, and EXPAND. The results were 3.05%, 1.84%, and 1.32%, respectively.

4. Discussion

Table 2 supports our hypothesis that the EXPAND interface would perform better than both the ASTER interface and the COLOR interface in the medium directory. The table shows that for several tasks, there were significant differences between ASTER and EXPAND, and COLOR and EXPAND, for the medium directory.

We were surprised at the lack of differences in performance between EXPAND and COLOR, and EXPAND and ASTER for Q7. We speculate that two circumstances might have caused this lack of differences. First, "umcp_tai" was on the second page of the "ls" listing (Appendix B). This would make it harder for the participants to locate the file immediately. Our observations during the experiment confirmed that some users had to spend several seconds before realizing that the file was not on the first page, since they were not told that the files were sorted in alphabetical order. Second, the file age slider provided in this experiment showed only the number of days starting from the day of the experiment. The slider did not show the date in the similar form as in "ls -l" listing (i. e., month/day/year). This idiosyncrasy made it hard for the participants to relate the numbers of days on the slider to the date displayed on the “ls -l” listing. This was also confirmed in some participants' comments which indicated the usefulness of this feature, if it had been provided.

It was also confirmed that ASTER and COLOR did not have significant performance difference both in the small and the medium directories. For Q8 through Q10, there was no significant difference among all three interfaces. One possible explanation will be provided in the next section.

4.1 Timed Tasks

<u>Task 1:</u> Some participants (6) did not use the slider to find the smallest file in the current directory. Instead, those who were familiar with the Unix environment just inspected the file size column in the file listing and picked out the smallest one. In some instances using visual inspection was faster than using sliders, especially for the small directory.

<u>Task 2:</u> For those who used visual inspection to answer the first question, the directories were so arranged that choosing the largest three files by visual inspection was difficult. From Table 2, we can see that EXPAND performed significantly faster than ASTER and COLOR for the medium listing.

<u>Task 3:</u> Most of the participants had mastered the skill of manipulating the sliders by this time in the experiment. They had also learned that there was no need to move the slider(s) to a position so that only the oldest file could be identified. Instead, the participants moved the age slider so that there were more than the one oldest file identified on the screen. They then visually inspected the selected entries to pick out the oldest file. From our experiences, this approach performed better than moving the age slider to the exact position where exactly one entry could be identified. This was due to the fact that the fine tuning control of OpenWindows' sliders required a significant amount of training.

<u>Task 4:</u> This task was similar to Task 2. EXPAND performed significantly better than ASTER and COLOR for the medium listing.

Task 5: Some participants commented that Task 5 was the easiest among the first five timed tasks. After the training session and the experiences gained from the first four tasks, all participants could move both age slider and size slider to the desired positions and then pushed the correct buttons without any difficulty. For the small directory, there was no performance difference among the three interfaces. Though the t-ratios did not reach significant levels for the medium directory, mean times for EXPAND were lower than for COLOR which in turn were lower than for ASTER.

Task 6: This task required the users to find out how many files were greater than "dq_group" in size. To answer this task, all users first located "dq_group" and found its file size. They then moved the size slider to that value and pushed the "greater than" button. For this task, the performance of the three interfaces also confirmed our hypothesis: EXPAND performed significantly better than both ASTER and COLOR in the medium directory.

Task 7: The correct way to solve Task 7, "How many files are there that are younger than 'umcp_tai'?", was to move the slider around and to pay attention to the file "umcp_tai." Users should stop the age slider when the file changed from de-selected to selected, or vice versa, depending upon which direction the slider was moving. The value displayed when the age slider stopped was the threshold. This threshold value should then be used to count the number of matched files. The statistical analysis did not show any significant difference among the three interfaces for this timed task.

Task 8 & 9: These tasks required the users to estimate the average file size in the current directory. Most of the users spent a great deal of time manipulating the size slider and its associated radio buttons, trying to find a good way to estimate. Only two of them used the correct trend finding technique when using the slider to estimate the file size at different size ranges. These numbers were then tallied (roughly) and divided by the number of the files to get the average size estimate. Most of the users gave up and provided a crude estimate after manipulating the size slider for a certain amount of time.

The same group of users quickly gave up (without using the age slider) and provided another crude file age estimate for task 9. From table 1, we can see that the time taken to finish Task 9 was much shorter than that of Task 8. Therefore we do not think the statistics on both Table 1 and Table 2 for Task 8 and 9 are meaningful.

Task 10: The last timed task was a multiple choice question. The task required the users to find out which size range held the majority of the files. Similar to Task 8 and Task 9, this was also a trend-finding task in which users were expected to move the size slider and to pay attention to the trend of the files that were selected. A fair number of articipants (11) counted the number of files as they moved the slider instead of using the trend-finding approach. The slow counting process was probably the reason why the statistical results did not show any significant difference among the three interfaces (Table 2) for this task.

4.2 Interface Characteristics

Studying the use of the slider revealed several interesting possibilities for improvements. Most subjects had never used the optical mouse before and had problems pointing accurately enough with it. Some subjects (3) commented that optical mouse did not give them the feeling of acceleration, as the mechanical mouse did. This unfamiliarity caused problems with the slider since the drag box of OpenWindows' slider is small. In a similar vein, several subjects found it hard to click on the even smaller slider bar to "fine tune" the setting. The fine tuning feature also caused problems as the mouse arrow moved to the end of the slider bar when users clicked on it. For some participants, moving the slider too fast seemed to cause confusion. Occasionally subjects were found clicking at the sides of the slider bar, to "fine tune" the slider up/down one step at a time, when they were supposed to make big changes. When asked, they commented that they were avoiding the sudden screen update.

4.3 Subjective Evaluation

Subjective ratings made for each interface were calculated and compared using the t-test method between any two interfaces of the three. Results from the t-test show that there were no significant differences between ASTER and COLOR, ASTER and EXPAND, or COLOR and EXPAND. We speculate the reason might be that every interface got relatively high scores such as at least 6, mostly 7s. The subjects might have been comparing these three interfaces with a traditional method, such as visual inspection with a "dir" command and therefore felt comfortable for all the three interfaces.

5. Research Directions

One might ask, "What is the advantage of employing dynamic query in a directory browser compared with the conventional directory browsing commands, such as dir or ls?" The main advantage of dynamic queries is that typical queries can be answered more rapidly because users can filter out irrelevant information and visually scan the remaining information. The rapid and continuous feedback of information helps the user to develop a clearer model of the age and size of the directory contents. When using the conventional command method, answering questions like "what are the file(s) with sizes larger than 15K in the current directory?" requires more time because users must visually scan a much larger set of information.

We conjecture that as the size of the directory grows the EXPAND interface will gain effectiveness, because the dynamic query approach will have even a greater effect in filtering the information.

Exactly how useful a dynamic query directory browser is in day-to-day application is yet to be explored. Other directions include testing with users with different levels of experience and testing with a wider range of tasks.

There are several issues that are worthy of further study. First, the handling of discrete selection criteria, such as owner, regular-expression-filtered file name, file access rights and property, and file type, should be different from the handling of continuous variables such as the file size. For continuous variables, the use of sliders is appropriate, while buttons are suitable for variables with two to ten values. The selection of file owners (i.e. login names in Unix's context) could be accomplished by a selector box that allows users to enter a regular expression that can be used as a pattern matching filter. As for the file access rights, a set of push buttons is appropriate. If a full implementation was contemplated, additional features such as clicking on a file name to view the contents, combining directories, and navigating among directories would be added.

The current dynamic queries interface uses conjunction (AND) to combine attributes but more complex boolean expressions (a common practice in database query) could be accommodated by applying new techniques of visual query formulation [8]. Further refinements to dynamic queries could come from more thoughtful application of color coding, use of sliders with two drag boxes to permit range queries, implementation of audio feedback to provide another medium, and novel highlighting techniques to show partial matches.

As direct manipulation graphical user interfaces (GUI) receive wider acceptance in the computer community, user interface designers can substantially improve current products by applying dynamic queries to make the file locating process and other tasks more efficient.

Acknowledgements

Some ideas in the dynamic query discussion were contributed by Richard Chimera, and Christopher Ahlberg, who was a Swedish visiting scholar at HCIL-University of Maryland. We also thank Wendy Cheng for her efforts in proofreading this article.

Bibliography

1. Shneiderman, B., "Direct Manipulation: A Step Beyond Programming Languages," *IEEE Computer*, Vol. 16, No. 8 (August 1983), pp. 57-69.

2. Larsson, J.A., "A Visual Approach to Browsing in a Database Environment," *IEEE Computer*, Vol. 19, No. 6 (June 1986), pp. 62-71.

3. Ahlberg, C., Williamson, C., and Shneiderman, B. "Dynamic Queries: Applying Direct Manipulation to Information Exploration," *Proceedings of ACM CHI'92 Human Factors in Computing Systems*, Monterey, CA (May 1992), pp. 619-626.

4. Williamson, C. and B. Shneiderman, "The Dynamic HomeFinder: Evaluating Dynamic Queries in a Real-Estate Information Exploration System," (to appear in *Proceedings of ACM IR conference,* Copenhagen, Denmark, June 1992).

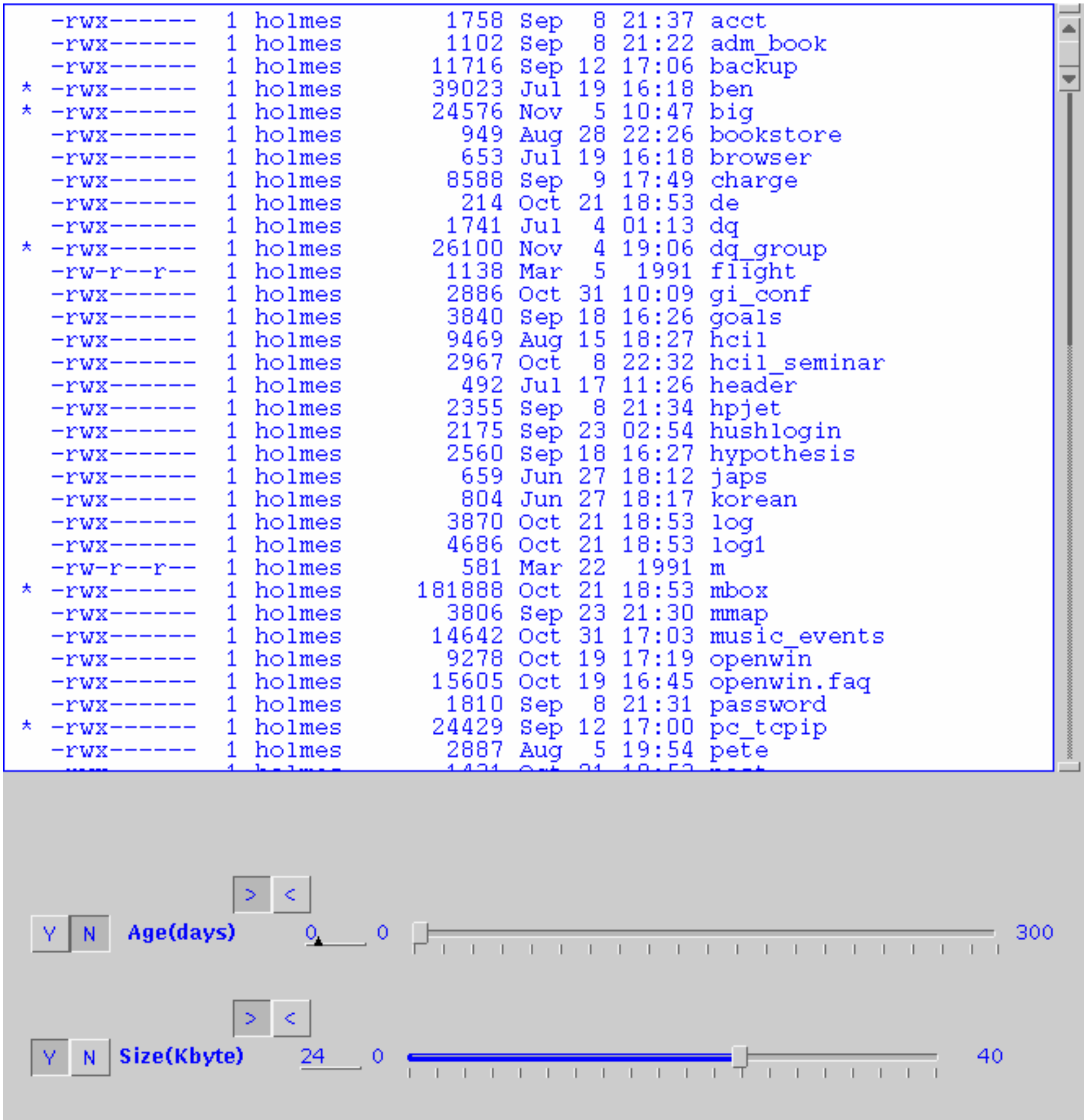
5. Akin, O., Baykan, C., and Rao, D.R., "Structure of a Directory Space: A Case Study with a Unix Operating System," *Int. J. Man-Machine Studies 26*, (1987) pp. 361-382.

6. Chin, J.P., Diehl, V.A., and Norman, K., "Development of an Instrument Measuring User Satisfaction of the Human-Computer Interface," *Proceedings of CHI'90 Human Factors in Computing Systems Conference*, ACM, New York, 1988.

7. Lee, E. and J. MacGregor, "Minimizing User Search Time in Menu Retrieval Systems," *Human Factors*, Vol. 27, No. 2, (1985) pp. 157-162.

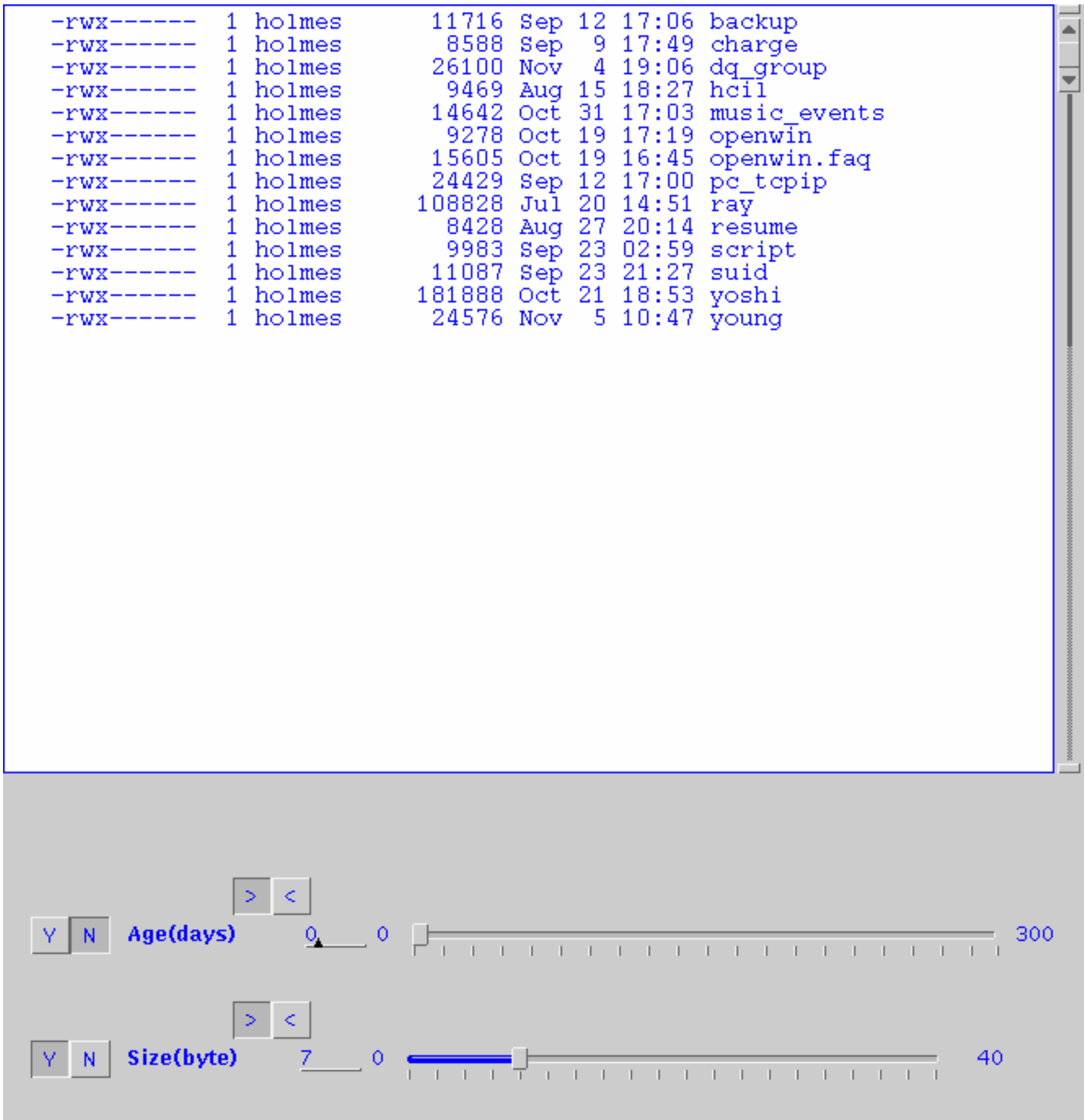
8. Shneiderman, B., "Visual User Interfaces for Information Exploration," *Proc. American Society for Information Science Conference*, Vol. 28 (1991), pp. 379-384.

Appendix A-1. Screen Image for ASTER


-rwx------ 1 holmes 1758 Sep 8 21:37 acct
-rwx------ 1 holmes 1102 Sep 8 21:22 adm_book
-rwx------ 1 holmes 11716 Sep 12 17:06 backup
* -rwx------ 1 holmes 39023 Jul 19 16:18 ben
* -rwx------ 1 holmes 24576 Nov 5 10:47 big
-rwx------ 1 holmes 949 Aug 28 22:26 bookstore
-rwx------ 1 holmes 653 Jul 19 16:18 browser
-rwx------ 1 holmes 8588 Sep 9 17:49 charge
-rwx------ 1 holmes 214 Oct 21 18:53 de
-rwx------ 1 holmes 1741 Jul 4 01:13 dq
* -rwx------ 1 holmes 26100 Nov 4 19:06 dq_group
-rw-r--r-- 1 holmes 1138 Mar 5 1991 flight
-rwx------ 1 holmes 2886 Oct 31 10:09 gi_conf
-rwx------ 1 holmes 3840 Sep 18 16:26 goals
-rwx------ 1 holmes 9469 Aug 15 18:27 hcil
-rwx------ 1 holmes 2967 Oct 8 22:32 hcil_seminar
-rwx------ 1 holmes 492 Jul 17 11:26 header
-rwx------ 1 holmes 2355 Sep 8 21:34 hpjet
-rwx------ 1 holmes 2175 Sep 23 02:54 hushlogin
-rwx------ 1 holmes 2560 Sep 18 16:27 hypothesis
-rwx------ 1 holmes 659 Jun 27 18:12 japs
-rwx------ 1 holmes 804 Jun 27 18:17 korean
-rwx------ 1 holmes 3870 Oct 21 18:53 log
-rwx------ 1 holmes 4686 Oct 21 18:53 log1
-rw-r--r-- 1 holmes 581 Mar 22 1991 m
* -rwx------ 1 holmes 181888 Oct 21 18:53 mbox
-rwx------ 1 holmes 3806 Sep 23 21:30 mmap
-rwx------ 1 holmes 14642 Oct 31 17:03 music_events
-rwx------ 1 holmes 9278 Oct 19 17:19 openwin
-rwx------ 1 holmes 15605 Oct 19 16:45 openwin.faq
-rwx------ 1 holmes 1810 Sep 8 21:31 password
* -rwx------ 1 holmes 24429 Sep 12 17:00 pc_tcpip
-rwx------ 1 holmes 2887 Aug 5 19:54 pete
> <
Y N Age(days) 0 0 300
> <
Y N Size(Kbyte) 24 0 40

Appendix A-2. Screen Image for COLOR

Appendix A-3. Screen Image for EXPAND



Appendix B. Timed Tasks

Q1. What is the smallest file in this directory?

Q2. What are the three largest files in this directory?

Q3. What is the oldest file in this directory?

Q4. What are the three youngest files in this directory?

Q5. How many files are there that are greater than 10 K in size and are older than 30 days in age?

Q6. How many files are there that are greater in size than "dq_group"?

Q7. How many files are there that are younger in age than "umcp_tai"?

Q8. What is your estimate of the average file size in this directory?

Q9. What is your estimate of the average file age in this directory?

Q10. In what size range do you think the majority of the files fall into?

[ ] 0 -1 K
[ ] 1 K - 5 K
[ ] 5 K - 10K
[ ] 10 K - 50 K
[ ] 50 K or above

Appendix C. Modified Version of QUIS

Overall reactions to the system:

1. terrible 1 2 3 4 5 6 7 8 9 wonderful
2. frustrating 1 2 3 4 5 6 7 8 9 satisfying
3. dull 1 2 3 4 5 6 7 8 9 stimulating
4. difficult 1 2 3 4 5 6 7 8 9 easy

Compare directory browser with other systems's directory commands (DOS "dir", VAX/VMS "dir", Macintosh Finder, UNIX "ls -l", etc.)

5. frustrating 1 2 3 4 5 6 7 8 9 satisfying
6. difficult 1 2 3 4 5 6 7 8 9 easy
7. useless 1 2 3 4 5 6 7 8 9 useful

User can control amount of feedback

8. never 1 2 3 4 5 6 7 8 9 always

Learning to operate the system

9. difficult 1 2 3 4 5 6 7 8 9 easy

Getting started

10. difficult 1 2 3 4 5 6 7 8 9 easy

Time to learn to use the system

11. slow 1 2 3 4 5 6 7 8 9 fast

Tasks can be performed in a straight-forward manner

12. never 1 2 3 4 5 6 7 8 9 always

Number of steps per task

13. too many 1 2 3 4 5 6 7 8 9 just right

Steps to complete a task follow a logical sequence

14. rarely 1 2 3 4 5 6 7 8 9 always

Completion of sequence of steps

15. unclear 1 2 3 4 5 6 7 8 9 clear

System speed

16. too slow 1 2 3 4 5 6 7 8 9 fast enough

Response time for most operations

17. too slow 1 2 3 4 5 6 7 8 9 fast enough

Correcting your mistakes

18. difficult 1 2 3 4 5 6 7 8 9 easy

Use of sliders in the program

19. difficult 1 2 3 4 5 6 7 8 9 easy