Evaluation of Text Editors

Teresa L. Roberts
Xerox Systems Development Department
3333 Coyote Hill Road
Palo Alto, CA 94304

Thomas P. Moran
Xerox Palo Alto Research Center
3333 Coyote Hill Road
Palo Alto, CA 94304

This paper presents a methodology for evaluating computer text editors from the viewpoint of their users—from novices learning the editor to dedicated experts who have mastered the editor. The dimensions which this methodology addresses are:

—*Time* to perform edit tasks by experts.
—*Errors* made by experts.
—*Learning* of basic edit tasks by novices.
—*Functionality* over all possible edit tasks.

The methodology is objective and thorough, yet easy to use. The criterion of *objectivity* implies that the evaluation scheme not be biased in favor of any particular editor's conceptual model—its way of representing text and operations on the text. In addition, data is gathered by observing people who are equally familiar with each system. *Thoroughness* implies that several different aspects of editor usage be considered. *Ease-of-use* means that the methodology is usable by editor designers, managers of word processing centers, or other non-psychologists who need this kind of information, but have limited time and equipment resources.

In this paper, we explain the methodology first, then give some interesting empirical results from applying it to several editors.

THE METHODOLOGY

The methodology is based on a taxonomy of 212 editing tasks which could be performed by a text editor. These tasks are specified in terms of their effect on a document, independent of any specific editor's conceptual model. The tasks cover:

—modifying the content of the document,
—altering the appearance of paragraphs and characters and the page layout,
—creating and modifying special kinds of text (such as tables),
—specifying locations and text in the document in various ways,
—programming automatic repetition of edits,
—displaying the document in various ways,
—printing, filing, and other miscellaneous tasks.

The functionality dimension of an editor is measured with respect to this taxonomy. However, comparisons between editors on the performance dimensions (time, errors, and learning) must be done on tasks which all editors can do. For this purpose, a set of 32 *core tasks* was identified.

The core tasks were chosen to be those tasks that most editors perform and that are most common in everyday work. Most of the core tasks are generated by crossing a set of basic text editing operations with a set of basic text entities. Thus, a core task consists of one of the operations (*insert, delete, replace, move, copy, transpose, split, merge*) applied to one of (or a string of) the text entities (*character, word, number, sentence, paragraph, line, section*). The core tasks also include *locating a place* in the online document which corresponds to a place in a hardcopy document (using the editor's simplest addressing mechanism), *locating a string of text* according to its contents, *displaying* a continuous segment of the document, *saving* and *retrieving* copies of the document, *printing*, and *creating* a new document.

Time. The speed at which normal text modification can be done is measured by observing expert users as they perform a set of *benchmark tasks* from the core tasks. There are 50 editing tasks in the benchmark, embedded in four documents: a short inter-office memo, two two-page reports, and one chapter from a philosophy book. The locations and complexities of the benchmark tasks are randomly distributed. The distribution emphasizes small tasks because those are most common in normal work and tasks involving boundary conditions in order to identify special cases, such as insertion at the beginning of a paragraph, which editors may treat awkwardly. Four

This paper is to appear in the *Proceedings of the Conference on Human Factors in Computer Systems*, Gaithersburg, Md., 15-17 March 1982, published by the Association for Computing Machinery.

experts are tested separately on the benchmarks. They are chosen to represent a spectrum of the user community: at least one user must be *non-technical* (i.e., does not have a programming background) and at least one must be *technical* (i.e., is very familiar with programming). The evaluator measures the performance in the test sessions with a stopwatch, timing the error-free performance of the tasks (errors are discussed below), and noting whether or not all tasks are performed correctly. This method of measurement is used because of the requirement that the test be easy for anyone to run (not everyone has an instrumented editor or a videotape setup, but anyone can acquire a stopwatch). That is also the reason for the limited number of subjects. The benchmark tasks typically take 30 minutes of steady work to complete. The score which results from this part of the test is the average error-free time to perform each task (the error-free time is the elapsed time minus time spent making and correcting errors). The overall time score is the average score for the four experts.

Additional information about the speed of use of a text editor may be obtained by applying the theoretical Keystroke-Level Model [1] to the benchmark tasks. This model predicts editing time by counting the number of physical and mental operations required to perform a task and by assigning a standard time to each operation. The operations counted include typing, pointing with the mouse, homing on the keyboard, mentally preparing for a group of physical operations, and waiting for system responses. In the present methodology, the evaluator must predict what methods a user would employ to perform the benchmark tasks; then the model is used to predict the amount of time to execute those methods. Differences between the conditions under which the Keystroke-Level Model was validated and the conditions here (e.g., small typographic errors are included, not all subjects use the same methods, etc.) lead to expected differences between predicted performance and the results of the experiments above. However, in addition to being a prediction of the benchmark time, the model also serves as a theoretical standard of expert performance.

Errors. The error-proneness of the editor is measured by recording the amount of time the expert users spend making and correcting errors on the benchmark tasks. Only those errors which take more than a few seconds to correct are noted (which is the best that can be done with a stopwatch). Thus, the time taken by simple typographical errors is not counted. Actually, this does not hurt the error time estimate too much, since the total amount of time in these kinds of small errors is relatively small. In addition to timing errors made and corrected while the user is working on the benchmarks, the evaluator also notes the tasks incorrectly performed; at the end of the experiment the user is asked to go back and complete those tasks correctly. The time to redo these tasks is added to the error time. Thus, the error score consists of all this error time as a percentage of the error-free time. The overall error score is the average for the four expert users.

Learning. The ease of learning to perform basic text modifications on the editor is tested by teaching four novices (with no previous computer or word processing experience) to perform the core tasks. The learning tests are performed in a one-on-one situation, i.e., by individually teaching each novice the editor. The evaluator orally teaches the novice how to do the core tasks in the editor, and the subject practices the tasks on the system. The methodology specifies the order in which to teach the tasks, but it is up to the evaluator to determine which specific editor commands to teach. Although all the teaching is oral, the evaluator supplies the novice with a one-page summary sheet listing all commands, so that the training is not hung up because of simple memory difficulties. After a set of tasks is taught, the novice is given a quiz, consisting of a document marked with changes to be made. Only a sample of possible tasks appears on each quiz, and not all tasks on the quiz have necessarily been taught up to that point. This allows for the novice to figure out, if possible, how to do tasks which haven't explicitly been taught. Referring to the summary sheet is permitted, but discouraged. The novice performs all of the tasks that he or she knows how to do, after which s/he is invited to take a short break if s/he wants it. Then another teaching period begins. In all, there are five training-plus-quiz cycles to teach all of the core tasks. Learning is evaluated by scoring the number of different tasks the subject has shown the ability to perform on the quizzes. The learning score is the total number of different tasks learned divided by the amount of time taken for the experiment, that is, the average time it takes to learn a task. The overall learning score is the average learning time for the four novices.

Functionality. The range of functionality available in the editor is tested by a checklist of tasks covering the full task taxonomy. Determining whether a task can be done or not with a given system isn't as trivial as it seems at first glance. Almost any task can be performed on almost any system, given enough effort. Consequently, the editor gets full credit for a task only if the task can be done efficiently with the system. It gets half credit if the task can be done clumsily (where clumsiness has several aspects: repetitiousness, requiring excessive text typing, limitations in the values of parameters to the task, interference with other functions, or a requirement of substantial planning by the user). The editor gets no credit for a task if either it can't be done at all (like use of italic typefaces on a system made for a line printer) or if doing the task requires as much work as retyping all affected text (such as manually inserting a heading on every page). The functionality checklist is filled out by a very experienced user of the editor, who may refer to a reference manual to ensure accuracy. The overall functionality score is the percentage of the total number of tasks that the editor can do. This percentage may be broken down by subclasses of tasks to show the strengths and weakness of the editor.

EMPIRICAL RESULTS

This methodology has been used to evaluate a diverse set of eight text editors: TECO [5], WYLBUR [9], a WANG word processor [10], NLS [3,4], EMACS [8], BRAVO [7], BRAVOX [6], and GYPSY (the last three editors are experimental systems developed within Xerox). The first two of these editors are made for teletype-like terminals, the rest are for display-based terminals. The intended users of these editors range from devoted system hackers to publishers and secretaries who have had little or no contact with computers. The results of these evaluations may be used in several ways: (1) as a comparison of the editors, (2) as a validation of the evaluation methodology itself, and (3) as general behavioral data on user performance.

Comparison of Editors. An editor's evaluation is a multi-dimensional score—a four-tuple of numbers, one from each performance dimension. A summary of the overall evaluation scores for the eight editors is given in Figure 1. Differences were found between the editors on all the evaluation dimensions (although only large differences were statistically significant, because of the large individual differences between the users tested). No editor was superior on all dimensions, indicating that tradeoffs must be made in deciding which editor is most appropriate for a given application.

	Evaluation Dimensions						
Editor	Time *M* ± *CV* (sec/task)		Errors *M* ± *CV* (% Time)		Learning *M* ± *CV* (min/task)		Functionality (% tasks)
TECO	49 ± .17		15% ± .70		19.5 ± .29		39%
WYLBUR	42 ± .15		18% ± .85		8.2 ± .24		42%
EMACS	37 ± .15		6% ± 1.2		6.6 ± .22		49%
NLS	29 ± .15		22% ± .71		7.7 ± .26		77%
BRAVOX	29 ± .29		8% ± 1.0		5.4 ± .08		70%
WANG	26 ± .21		11% ± 1.1		6.2 ± .45		50%
BRAVO	26 ± .32		8% ± .75		7.3 ± .14		59%
GYPSY	19 ± .11		4% ± 2.1		4.3 ± .26		37%
M(M) M(CV)	32	.19	11%	1.0	8.1	.24	53%
CV(M)	.30		.54		.58		.28

Figure 1. Overall evaluation scores for eight text-editors.
The Time score is the average error-free time per benchmark task. The Error score is the average time, as a percentage of the error-free, that time experts spend making and correcting errors. The Learning score is the average time to learn a core task. The Functionality score is the percentage of the tasks in the task taxonomy that can be accomplished with the editor. The *Coefficient of Variation (CV) = Standard Deviation / Mean* is a normalized measure of variability. The *CV*'s on the individual scores indicate the amount of between-user variability. The *M(CV)*'s give the mean between-user variability on each dimension, and the *CV(M)*'s give the mean between-editor variance on each dimension. The evaluations for TECO, NLS, WANG, and WYLBUR are from Roberts [2].

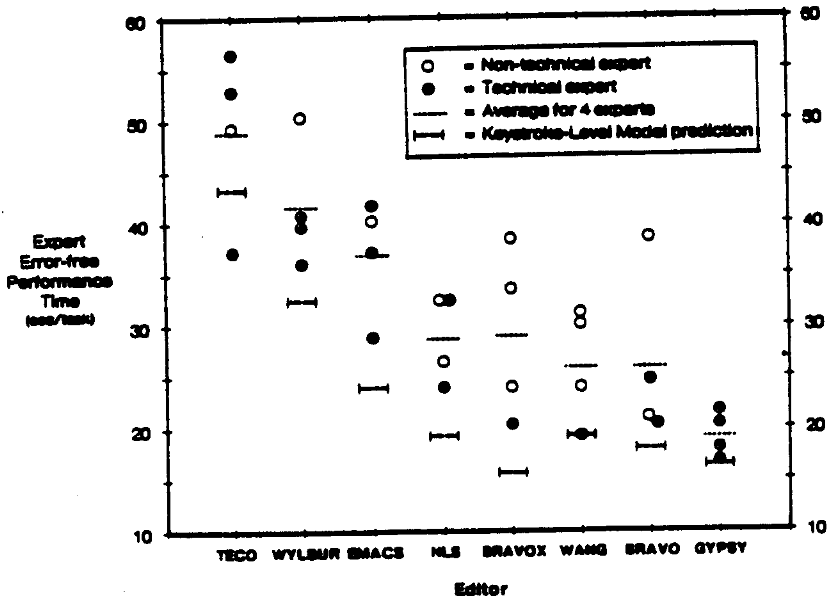

Figure 2. Time scores for individual expert users.

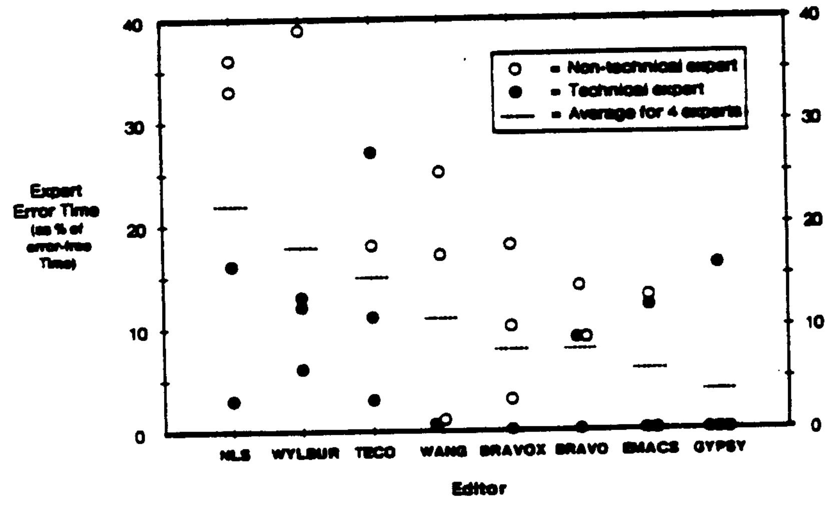

Figure 3. Error scores for individual expert users.

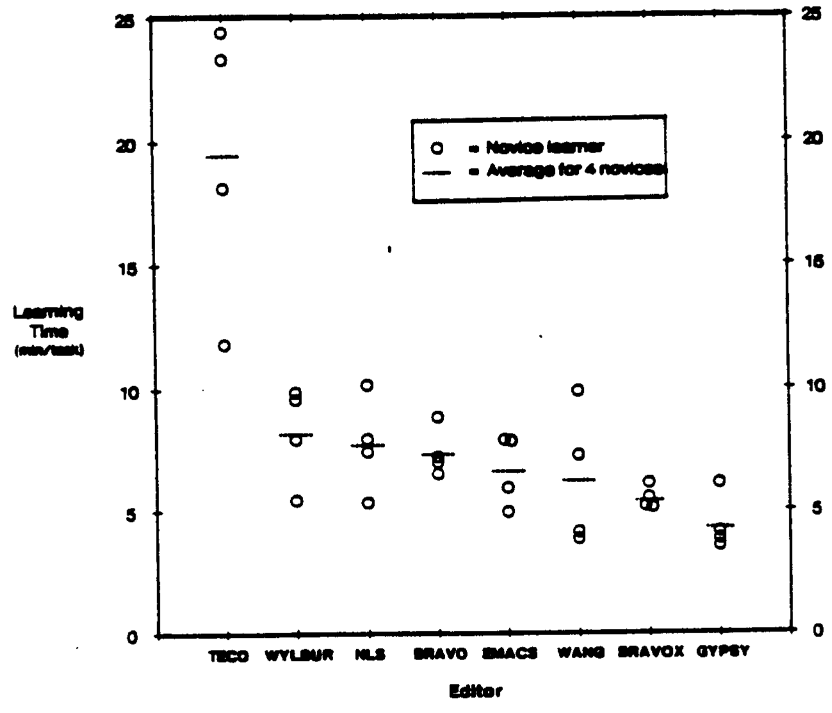

Figure 4. Learning scores for individual novice learners.

Time. The summary time data in Figure 1 is expanded in Figure 2 to show individual users' scores. These results show that TECO, WYLBUR, and EMACS are the slowest editors and that GYPSY is the fastest. Most of the display-based systems were used at about twice the speed of the non-display systems.

The times predicted by the Keystroke-Level Model, also shown in Figure 2, can be seen to be about 70% of the average error-free experimental time. But the data for individual users show that for most editors, one user comes very close to the "standard expert" performance that the Keystroke-Level Model predicts.

Errors. Individual users' error scores are shown in Figure 3. This data finds a factor of five difference between the best and the worst editors, but even so these differences are small compared to the differences between users. No conclusions about the error-proneness of editors can be drawn.

Learning. The overall learning scores are shown in Figure 1, and the scores for individual novices are shown in Figure 4. Large differences were found in the learnability of the different editors. TECO turned out to be an outlier, taking over twice as long to learn as the next editor (WYLBUR). The rest of the editors lie along a smooth progression which covers another factor of two in learning rate, with GYPSY being four times faster to learn than TECO.

Learning/Speed Tradeoff. The conventional wisdom is that there is a tradeoff between the speed of learning and the speed of expert use of a system. Combining the learning results with the time results, we see exactly the opposite. The data from this study shows a high positive correlation ($R = .80$) between the time and learning scores. The major difference was between the set of display editors and the set of non-display editors. Display editors are better for both expert time and novice learning.

Functionality. The functionality results showed that most of the editors could perform roughly half of the tasks in the task taxonomy. The reason for this was that, in general, each system had its areas of strength and weakness. In order to show this, the data are broken down by task categories in Figure 5. EMACS shows up well in programming capability, while NLS and BRAVOX shine in formatting and layout. Because the numbers of tasks in the taxonomy was more weighted toward text layout than editor programming, the document-oriented editors generally showed up as being somewhat better than EMACS. But NLS, which tries to cover all needs, was the clear winner in overall functionality.

Evaluation of the Methodology. The results above show that diverse editors can indeed be evaluated and compared. As a whole, the evaluation methodology seems to successfully provide an objective, multi-dimensional picture of text editors. This methodology is also quite practical. For an experienced evaluator, about one week of time is required to evaluate a new editor; thus it is quite accessible to a system designer or a potential buyer.

There are still many areas of editor use which are not covered by the methodology, for instance the needs of occasional users. In addition, use of the methodology has pointed out areas, such as error-proneness, where more reliable measures are needed to differentiate specific editors. Any further work will have to take into account the effect of large differences among the users.

Behavioral Results. The data gathered from these evaluations are also interesting for what they tell us about user behavior. It provides a pool of data on overall levels of user performance. We see from this data that typical core editing tasks require on the order of 30 seconds for experts to perform, and we see that a period of about two hours of one-on-one training is enough to get users started with the core tasks in most editors. Such data should be of interest to researchers in office productivity, e.g., to measure the cost-effectiveness of word processing.

Our data also provides some insight into individual differences in performance among users:

(1) By far the greatest individual differences are found in error rate (ranging from 0% to 39%), which reflects a wide variation in the style of using text editors.

(2) Comparing speed of expert, error-free use with error rate shows that it is the slower users who make more errors. So while there may be a speed/error tradeoff that an individual user can make, we don't see such a tradeoff between people.

(3) There is only moderate variation among experts in speed of editing—about a factor of 1.5 to 2 between the fastest and slowest user within each editor.

(4) A somewhat surprising result is that there is not much more variation among novice learners than among experts, i.e., about the same range of differences between the fastest and slowest learners.

Conclusion. This methodology has proven itself to be an effective tool for the empirical evaluation of text editors, helping us understand how people adapt to them as well as providing us with much-needed feedback on how they actually perform.

Task Category	# of Tasks	Editor								All Editors
		NLS	BRAVOX	BRAVO	EMACS	WANG	WYLBUR	TECO	GYPSY	*M±CV*
TOTAL	*212*	77	70	59	49	48	42	39	37	53±.28
Modification										
Content	*66*	94	89	90	74	87	63	88	80	83±.12
Text Layout	*19*	89	71	71	37	37	26	3	26	45±.65
Page Layout	*25*	74	62	40	2	34	6	4	4	28±1.0
Characters	*21*	43	76	62	14	38	21	0	17	34±.76
Special Purpose	*16*	53	59	22	0	34	16	3	0	23±1.0
Addressing	*22*	68	36	30	61	16	34	25	18	43±.60
Control	*23*	56	37	20	89	24	61	48	9	64±.39
Display	*8*	94	94	69	81	19	62	38	50	63±.42
Misc.	*12*	100	88	71	46	71	71	25	42	36±.53

Figure 5. Functionality sub-scores for eight text-editors.
Each functionality sub-score is given as a percentage of the total number of tasks in its task category (the italic numbers in the "# of Tasks" column). The numbers in the "All Editors" column summarize how well the task categories are handled by the whole collection of editors.

Acknowledgement. We would like to thank Betsey Summers for organizing and running many of these evaluation studies and for helping us analyze the data.

REFERENCES

[1] Card, S. K., Moran, T. P., and Newell A. The Keystroke-Level Model for user performance time with interactive systems. *Communications of the ACM*, 1980, *23*, 396-410.

[2] Roberts, T. L. *Evaluation of Computer Text Editors.* Ph.D. dissertation, Department of Computer Science, Stanford University, 1980. Available as Report AAD 80-11699 from University Microfilms, Ann Arbor, Michigan.

Editor Documentation References

[3] Augmentation Research Center. *NLS-8 Command Summary.* Menlo Park, California: Stanford Research Institute, May 1975.

[4] Augmentation Research Center. *NLS-8 Glossary.* Menlo Park, California: Stanford Research Institute, July 1975.

[5] Bolt, Beranek, and Newman, Inc. *TENEX Text Editor and Corrector* (Manual DEC10-NGZEB-D). Cambridge, Massachusetts: Author, 1973.

[6] Garcia, Karla. *Xerox Document System Reference Manual.* Palo Alto, California: Xerox Office Products Division, 1980.

[7] Palo Alto Research Center. *Alto User's Handbook.* Palo Alto, California: Xerox PARC, September 1979.

[8] Stallman, R. M. *EMACS Manual for ITS Users.* MIT, AI Lab Memo 554, 1980.

[9] Stanford Center for Information Processing. *Wylbur/370 The Stanford Timesharing System Reference Manual, 3rd ed.* Stanford, California: Stanford University, November 1975.

[10] Wang Laboratories, Inc. *Wang Word Processor Operator's Guide, 3rd release.* Lowell, Mass., 1978.