Human Aspects of Computing

Henry F. Ledgard Editor

Training Wheels in a User Interface

JOHN M. CARROLL and CAROLINE CARRITHERS

ABSTRACT: New users of high-function application systems can become frustrated and confused by the errors they make in the early stages of learning. A training interface for a commercial word processor was designed to make typical and troublesome error states "unreachable," thus eliminating the sources of some new-user learning problems. Creating a training environment from the basic function of the system itself afforded substantially faster learning coupled with better learning achievement and better performance on a comprehension post-test. A control group spent almost a quarter of their time recovering from the error states that the training interface blocked off. We speculate on how this training strategy might be refined, and more generally, on how function should be organized in a user interface.

Empirical characterizations of computer novices learning to use application systems paint a dreary picture of side tracks and error tangles from which recovery, or even diagnosis, is difficult (e.g., [6]). In this paper, we show that a "training wheels" interface—designed to block typical side tracks and error states—can facilitate the learning process for new users.

THE TRAINING WHEELS INTERFACE

We studied a stand-alone, commercial word processing system and observed new users, people who had never used a computer before, trying to learn the system's basic function. From this, we developed an inventory of typical new-user errors. The system's interface was then modified to block these error states, that is, to make them unreachable. This training wheels interface afforded the basic function for document creation, revision, and printing, but no advanced word processing function (e.g., table formatting). A variety of specific errors were identified and blocked. The most important (in terms of the severity of their consequences) were these seven (the errors are numbered and labeled for later reference):

1. ***The Exotic Menu Choice Error.*** New users often recklessly tried out menu choices in their early encounters with the system. These users typically became tangled in advanced menus (for example, those pertaining to system and diskette maintenance) and were unable to get back to the simple function they had originally set out to learn. This error was eliminated by making only basic menu choices accessible in the training wheels interface. When inappropriate, exotic choices were selected, the message "`XXX not available on the Training System`" was displayed (where "`XXX`" stands for the name of the selected function), and the advanced function was not engaged.

2. ***The Print First Error.*** New users want to undertake real work immediately. We observed novices who requested a print job before they had even created and stored a document. This error was

This paper was presented at the 24th Annual Meeting of the Psychonomic Society, on November 17, 1983, in San Diego, California.

blocked by making print requests dependent on immediately preceding create or revise requests. When a print was selected inappropriately, the message "`Print is available only after a Create or a Revise on the Training System`" was displayed.

3. ***The Parameter Loop Error.*** Several of the word processor's menus involve a series of parameters. Users have the option of accepting default values by merely pressing ENTER or of interacting with the menu to specify or respecify parameter values (for example, the default of single spacing can be respecified to double spacing). We observed that new users failed to avail themselves of the default option. They quite typically got into a loop of specification and respecification of parameters (perhaps because the screen prompt inviting this action was more visually salient than was the prompt inviting the defaults). These parameter loops can be lengthy and frustrating, and they are totally unnecessary. For the novice, the defaults are fine. This error state was blocked by disabling the menu parameters; when a novice selected a parameter, the disablement message "`XXX not available on the Training System`" was displayed (where "XXX" stands for the name of the selected parameter).
4. ***The Alternate Shift Error.*** We observed an error involving the alternation of two keystroke commands. Both are located on the same physical key. One command, which cancels menus and selections, requires an alternate shift. The other, which displays a special system utility menu, does not. Understandably, it is the alternate key that is more important to novice users (who must frequently cancel an incorrect menu or selection). However, these users often fail to hold down the alternate shift and end up with the utility menu offering seemingly inscrutable choices. This error state was blocked by disabling the nonalternated command. Pressing the key without the alternate shift elicited the message "`The Utility key is not available on the Training System.`"
5. ***The Print Queue Error.*** The word processor we studied allows multiple print jobs to be queued. Novices lack the concept of a "queue" and often issue multiple print requests. This leads to problems when the user later requests another print of another document. The output will be what was queued before and not what was just requested. The user is also prevented from revising documents that are queued to print, hence multiple print requests effectively make a document unrevisable. The error state was blocked by limiting the size of the print queue to one document. Trying to queue a second document merely elicited the message "`Only one document at a time can be printed on the Training System.`"
6. ***The Diskette Name Error.*** The word processor stores data on diskettes. In the course of creating, revising, and printing a document, the user is prompted to specify a diskette name, and then to insert that diskette. If the user misspecifies the diskette name, the system prompts for the (perhaps nonexistent) diskette. There is a command to cancel this request, but it is not an elementary command. For novices, this error is a dead end. This error state was eliminated by anticipating the name of the diskette that novices use and by programming the system to accept only that name. Hence, misspecifying the diskette name elicited the message "`Diskette XXX is not available on the Training System.`"
7. ***The Unprintable Character Error.*** Different printer configurations for the word processor are compatible with different subsets of its possible keyboard characters. In particular, for the printer we used, it was possible to enter keyboard characters into a text file which could not be printed. In such a case, the word processor produced an error message which prompted the user to override the printer, and have an underscore print in place of the unprintable character, or to cancel the print job altogether (and then, by implication, take further action, such as re-editing the document or changing the printer). This error state was eliminated by forcing a choice on the user: Unprintable characters were always replaced with underscores.

There were of course a variety of other possible errors users could make. To contrast them terminologically with the "blocked" errors above, we will refer to these as "nonblocked" errors. We will comment on some of these later.

Our hypothesis was that blocking the consequences of these seven errors would eliminate some of the frustration and confusion caused in the early stages of learning, and thereby facilitate learning of the system by novices. We performed an exploratory experimental investigation to test this hypothesis.

METHOD AND RESULTS

Our experimental approach was essentially observational, reflecting our concern with the qualitative differences between two training experiences. However, the observer in our study did *not* interfere with behavior (e.g., by questioning or prompting), and hence we will report sheer performance differences as well.

In the experiment, 12 office temporaries learned to use a commercial word processing system. They were obtained from an agency and selected to have no prior word-processing experience. At the start, they were each given a brief self-study manual and a handwritten letter. They were told to imagine that they had been hired to replace a typist for one week, were to learn the essentials of the word processing system, and then type up the letter—all this as quickly as possible. They were

asked to further imagine that the only person who knew about the word processor was the one they were replacing—hence, they would be totally on their own. The observer offered encouragement whenever it seemed appropriate, but the only "help" provided was the suggestion to reread the manual.

Six of the temporaries were arbitrarily assigned to the training wheels interface and six to the complete system interface. The manual they used was specially designed to be brief. (In 12 pages, it introduced word processing and taught system initialization, document creation, and printing.) The training wheels version of the manual differed from the complete system version only where it had to be factually accurate in order to inform the user of the disablement messages.

The observer sat with the participant during the entire session, taking detailed notes, including timings on actions and outcomes. Particular attention was given to errors (like 1-7 above)—both to their frequency and to the amount of time required to recover from them. ("Error" was defined as a departure from the create and print action path; "recovery" was defined as a subsequent return to that path).

Performance Differences

Overall time and success data indicate that the training wheels interface provided a better learning environment for our participants. Training wheels users took an average of 92 minutes to complete the letter-typing task; complete system users took 116 minutes. Thus, the training wheels participants were 21 percent faster overall, $t(10) = 2.11$, $p < 0.05$. (See [5] for description of t statistics.) Participants who failed to type and/or print out were arbitrarily assigned a time of 120 minutes. As we will see, most of these failures were in the complete system condition, hence the time analysis may be somewhat conservative. The performance, error and post-test measures we took are summarized in Table I.

Overall success followed the same pattern. Four of the six participants using the training wheels interface succeeded in printing out the letter we gave them within the allotted time. (Mean time to complete the task for these four people was 78 minutes.) The other two typed the letter into the system by the end of the two hours but did not print. In contrast, only two of the six participants using the complete system succeeded in printing their letters. (Mean time for these two was 107 minutes, 37 percent longer than the four training wheels interface participants who printed.) Two other participants had only typed the letter after two hours; two had neither typed nor printed their letters.

We performed Wilcoxon's ranked sum test [5] on the overall success data, grouping the participants into three classes—those that printed, those that only typed, and those that did neither. The significance level for this contrast was only marginal, $p < 0.1$. If we also include performance time information in this ranking, that is, if we rank by overall time within the three success classes, the Wilcoxon ranked sum becomes statistically significant, $p < 0.05$.

TABLE I. Performance, Error and Post-test Summary

	Training Wheels	Complete System
Overall Measures		
Total time on task	92 min.	116 min.
Number of subjects who printed	4	2
Number of subjects who only typed	2	2
Number of subjects who failed	0	2
Initial time to get to the typing area	28 min.	53.6 min., $n = 5$
Total time spent in the typing area	14.7 min.	10.2 min.
Total time spent on error recovery	45.5 min.	65 min.
Total recovery time for blocked errors	5.7 min.	25.2 min.
Learning versus Letter-Typing Phase:		
Total time spent in learning phase	75 min.	88.5 min.
Total time spent in letter-typing phase	16.8 min.	40.5 min., $n = 4$
Initial time to typing area in letter phase	3.5 min.	10.0 min., $n = 3$
Error recovery time in learning phase	39 min.	47.8 min.
Error recovery time in letter-typing phase	6.5 min.	25.8 min., $n = 4$
Post-tests		
Comprehension scores	2.7/6	1.6/6
Work-Attitude scores	12.5/16	9.8/16

In brief, these overall quantitative measures indicate that the training wheels interface afforded more efficient learning progress. Indeed, a third of the complete system participants were unable to produce any demonstration of practical competence after two hours of training time. The people using the training wheels interface were all able to do something, and the majority of them were able to complete the entire task. In passing, it is also worth noting that—looking across both conditions—half of our participants did not succeed in printing a one-page letter within two hours. This remains a major design challenge in word processing.

Beyond the overall differences, we wanted to understand what the training wheels learners were doing differently than the complete system learners. Most of the modifications in the training interface are intended to help the new user get through the control structure of the word processor and on to the more concrete tasks of creating and printing. Accordingly, one further performance measure we focused on was the time elapsed when the user first emerged from this control structure and accessed the typing display. The mean time to reach the typing area for the users in the training wheels condition was 28 minutes; for those using the complete system, the mean time was 53.6 minutes. The training wheels learners were 48 percent faster, $t(10) = 2.47$, $p < 0.05$.

Traversing the system's control structure to get to the typing display was indeed an important aspect to focus attention on. Across all 12 participants, only 12 percent of the 2-hour experimental session was spent working on the typing display. The fact that the training wheels participants got through this control structure more quickly may have left them relatively more time to work on the typing display. Training wheels participants spent an average of 14.7 minutes, or 16.5 percent of their time, in the typing display; the complete system participants spent 10.2 minutes, or 9 percent of their time, $t(10) = 2.97$, $p < 0.01$.

How were the training wheels learners able to traverse the system's control structure so much more rapidly? At least a part of the performance difference between the two groups can be directly ascribed to the amount of time the complete system learners wasted recovering from the particular errors (1–7 above) that the training wheels learners were protected against. Indeed, the complete systems learners spent almost 22 percent of their time on task (25.2 minutes) trying to recover from these seven particular errors. The training wheels learners were blocked from suffering the consequences of these errors and spent an average of less than 6 minutes (7 percent of their overall time on task) recovering from the blocked errors. (Generally, this was time spent reading and interpreting the training interface's disablement messages.) This difference in error recovery times between the two groups was significant, $t(10) = 3.20$, $p < 0.005$.

Again, we must keep this result in perspective. Both groups spent a huge amount of time recovering from errors. Learners using the training interface spent 50 percent of their time in error recovery as compared to those using the complete system who spent 56 percent of their time. (The training wheels learners were of course recovering from errors other than 1–7 above, the nonblocked errors).

The summary figures for time spent on error-recovery suggest a simple additive model differentiating the two learning conditions. It turned out that learners in both groups spent an average of about 40 minutes recovering from nonblocked errors, but that the complete system learners spent almost 20 minutes more, on average, recovering from the seven errors blocked for the training wheels learners. Recall now that the training wheels group spent about 24 minutes less on the overall letter-typing task; it seems that one could summarize the effect of the training interface as saving the learner 20 minutes by blocking 20 minutes worth of error-recovery time.

We do not believe, however, that the training interface merely saved people from making errors. There are several sources of evidence for this. First, there is evidence in the overall times that participants using the training interface enjoyed most of their advantage when they were actually performing with the system, as opposed to learning it. We were able to divide the participants' overall time into two phases—time spent initially learning, and time spent typing and printing the letter. We did not ask participants to organize their time into two sequenced phases, but they all did so. Training-wheels learners spent an average of 75 minutes in the learning phase and nearly 17 minutes in the letter phase. Only four of the complete system learners entered both phases; for these four an average of nearly 75 minutes was spent in the learning phase but over 40 minutes in the letter phase. Clearly, the time advantage of the training wheels group was in the letter phase of the experiment. This difference was significant, $t(8) = 2.90$, $p < 0.01$.

In order to separate the fact that training wheels learners spent less time on the experimental task overall from the contrast of time spent on the final letter phase, we recomputed the contrast using the proportion of each participant's time spent on the final letter phase. The difference is still reliable by Wilcoxon's ranked sums test, $p < 0.05$. Hence, the training wheels learners not only spent significantly less time on the final letter phase of the task, they spent a smaller proportion of their time on this phase of the task. These differences were obtained despite the fact that both groups spent comparable amounts of time in the initial learning phase. This indicates that the training wheels people were able to make better use of their learning time.

But what were they learning? Other performance evidence indicates that the training interface participants learned to avoid errors. Considering both types of errors (blocked and nonblocked), the training wheels learners spent 52 percent of their time, or an average of 39 minutes, recovering from errors during the learning phase of the experiment. The complete system people spent 54 percent of their time, an average of 48 minutes, in error recovery during the learning phase. In the letter phase, the training wheels group spent only 39 percent of their time, an average of less than 7 minutes on error recovery, while the complete system group (of only four participants) spent 64 percent of their time, 26 minutes on average. Thus, the training interface significantly reduced the proportion of time dedicated to error recovery between the learning phase and the letter phase relative to the complete system group, $t(8) = 2.1$, $p < 0.05$.

When we further divided these error recovery times into blocked versus nonblocked errors, we found that most (over 90 percent) of the recovery time decrease for the training wheels group came from the nonblocked errors. This is not entirely surprising; the training wheels people spend so little time recovering from the blocked errors that there was little room left for improvement. However, their improvement in the nonblocked errors suggested that blocking of even *some* learner errors can have a generalized benefit for learners.

Training wheels learners also seemed to become efficient users more rapidly. In the letter-typing phase, the training wheels people were able to traverse the system control structure and get to the typing area in an average of 3.5 minutes. Complete system learners took 10.0 minutes or more than three times as long, $t(8) = 3.69$,

$p < 0.005$. Between the learning phase and the letter phase, the training wheels learners reduced the amount of time it took them to get to the typing area by a factor of 8 (3.5 versus 28 minutes); the complete system learners reduced their times by a factor of 5 (10.0 versus 53.6 minutes). While both groups were improving, the training wheels group improved more dramatically.

Finally, two post-tests we administered also indicated that the training wheels manipulation had done more than merely limit error recovery time. We administered a brief comprehension test which asked participants in both groups about diskettes and documents, and about several of the function keys and display symbols. The results are an indication that the group on the training wheels did learn more than the group on the complete system. Mean number correct for the group using the training wheels was 2.7 out of 6; for the group using the complete system the mean number correct was 1.6. The training wheels learners did 41 percent better. This result is significant by Wilcoxon's ranked sum test, $p < 0.05$.

After the experiment proper, we also gave all participants a questionnaire designed to reveal their general attitude toward work. The group using the training wheels scored significantly higher than the group using the complete system, $t(10) = 2.32$, $p < 0.05$. The participants using the training interface, because they were more successful, may have felt better about themselves and about work in general.

Individual Differences

We observed vastly differing individual learning styles among the participants in our study. At one extreme is the reckless explorer who immediately begins to play with the system, frequently with only a superficial reading of the manual. This type of learner commits many errors and spends much of the time in error recovery, but will sometimes stumble on the correct solution by mistake. One of our participants made 106 errors during the two-hour test period.

At the other end of the continuum is the plodder who will not try anything until assured of the results. This type of learner will read and reread the manual and frequently sit and stare at the screen for periods of time. These participants make fewer errors but may have more trouble recovering from the effects of errors. One of our participants made only 15 errors total in two hours, but the fifteenth error was fatal and this person was unable to accomplish anything more in the final 16 minutes of the experimental session. Neither of these learning styles seems to be better-suited than the other to mastering the system. The two learners in these examples were equally successful; both were able to type in the letter, but neither was able to print.

One particular learner in the training wheels group caught our attention. All by herself, she accounted for nearly half of the errors committed by the learners in this group. Indeed, looking at the nonblocked errors made during the letter phase, this participant made 36 of the group's 43 errors. Not surprisingly, removing her from the experiment strengthened most of the statistical results we have cited. Curiously, she is the reckless explorer described above. (The plodder was one of our complete system participants.) Unfortunately, at present we have no independent basis for understanding these individual differences, and their causal relations to training methods and performance success.

Error Specifics

Examining the particular errors and error recovery outcomes can further illuminate how the training interface worked. First though, we digress to comment on a few of the principal "nonblocked" errors. Mechanical errors involving the loading of program or data diskettes consumed the most time among the nonblocked errors. A typical error of this sort was attempting to mount a diskette in an improper orientation. A second type of nonblocked error—with an assortment of variants—was naming errors, producing names with too many characters, naming a document with the same name as a menu choice, an ID letter, or a data diskette or using a double name (ignoring a system-prompted name and suffixing a duplication of the name). Another nonblocked error was making an overly literal response to a system prompt, for example, typing the literal string "`ID letter`" when prompted by the system to "`type ID letter`" (the system accepts only ID letters like a, b, c, etc.).

Of the seven errors inventoried in previous studies of the word processing system, and blocked in the training interface, four did not affect the performance of participants in either experimental condition. Only three participants added more than one document to the print queue and since they were all from the training wheels group they received the message that this was not allowed (Print Queue Error, #5). And only two participants typed characters that could not be printed (Unprintable Character Error, #7); both were in the training wheels condition and neither even noticed the underscores that the system substituted. The fact that all of these errors occurred in the training wheels group seemed likely to be due to the fact that only two of the complete system learners ever printed anything. Overall, the training wheels group printed nine documents to three for the complete system group.

Two other blocked errors also were of little importance in participants' performance. No participant attempted to print before creating or revising (Print First Error, #2), and only three participants made an exotic menu choice unrelated to their task, but were able to recover quickly (Exotic Menu Choice Error, #1). Two accounts of this are obvious. First, we prepared our own manual for this experiment—chiefly to control "manual effects" between the two system conditions. We designed the manual to be usable, based on what we had seen in the course of our earlier work and perhaps we simply succeeded with respect to Errors 1 and 2.

A second account appealed to our learners' task orientation. Both groups were explicitly instructed to type out a particular letter. Thus for them, learning the sys-

tem was a functional means of accomplishing as quickly as possible a real task they could already appreciate. Learners in our previous studies, and indeed typically in such studies, were instructed to "learn how to use the system." This relatively abstract goal may make learners more susceptible to certain errors. These possibilities are being pursued in follow-up research.

The other three blocked errors in our inventory account for the majority of the performance differences between our two groups. Five of the 12 participants specified unnecessary parameters in menus (Parameter Loop Error, #3). Of these five, four were in the complete system group and one in the training wheels group. One of these complete system participants seemed different from the others who made this error: She made only Error 3 on one menu, made the error only twice in all, and was the only person to make the error *after* having successfully avoided it earlier in the experiment. The other four leaners who made this error made it repeatedly and in different menu environments.

We cannot say that the training wheels interface helped our new users learn not to make Error 3 because this error seems to be self-sustaining once it is committed. However, the raw figures do suggest that people using the training interface are less likely to make the error in the first place. Perhaps, having already seen that many menu choices and keypress commands are disabled in the training interface, these new users are disinclined to go beyond the training manual instructions by trying a nonrequired menu option.

Although our learner sample is quite small, some preliminary observations seem striking. No participant in either group *clearly* learned *not* to make Error 3. In fact, in the only unequivocal case of learning, a participant learned how to *recover* quickly; this did not prevent her from making the error. The participants' first encounter with a potential error situation seemed to determine performance in future situations: Those participants who committed the error at their first opportunity always seemed to repeat it; those participants who did not commit it at the first opportunity, tended not to commit it at all.

Seven of the 12 participants made the Alternate Shift Error (#4), improperly using the alternate shift key when attempting to cancel menus or selections. Three of these seven used the training wheels interface; four used the complete system. In contrast to Error 3, five of the seven participants who made this error *did* demonstrate that they had learned to correct this mistake. This included all three of the training wheels learners and two of the complete system learners. One of the remaining participants used the cancel function correctly for a time, then began to make the error. The remaining participant never correctly used it, although she kept trying.

Error 4 was fairly pervasive; four of the five participants who avoided the error simply had no occasion to use the cancel function and therefore never risked making the error. Indeed, only one of the participants who tried the operation at all used it successfully throughout. Perhaps we do have here an indication that the training interface is supporting successful learning. Learners using both systems made Error 4, but all of the training wheels learners corrected the error where only half of the complete system learners were able to do this.

By far the most costly error for participants was misnaming a data diskette (Diskette Name Error, #6). It consumed large spans of learning time and it was fairly common; seven of the 12 learners made the error. Three of these were using the training wheels interface, four were using the complete system. Of these seven, only two seemed to have learned the correct concept by the end of the experiment. Both were in the training wheels group. Three of the remaining participants, one in the training wheels group, two on the complete system, managed to make progress in spite of the fact that they seemed to have no clear concept for a data diskette name. The last two participants, both on the complete system, were never able to use the diskette name correctly and were unable to continue making progress once they had committed the error. One learner spent 90 minutes attempting to correctly name a data diskette. Again, in spite of the small numbers, it is worth noting that the training interface fostered the learning that did occur; participants who learned to correctly name diskettes were using the training wheels interface while the participants who were unable to recover in the specified time were on the complete system.

DISCUSSION

The training system tested in this study presents a first approximation of a solution for introducing new users to the mechanics of complex systems. One obvious next step in this research would be to reiterate our selection of blocked errors. Errors we did not block (e.g., naming a document with the same name as that of the data diskette, and then confusing the two) turned out to be troublesome for our participants. Conversely, four of the seven errors whose consequences we did block, actually had little effect on the performance of participants. Another direction for research would be a closer examination of the disablement messages, which informed users that a requested system function was unavailable. In the training wheels interface, Error 4—neglecting to hold down the alternate shift while cancelling a menu or selection—elicited a message that the utility key was not available. This might be understandably confusing in that the person may never have intended to request the system utility menu in the first place! A better message might inform the person that it is necessary to hold down the alternate shift key while cancelling.

Beyond these technical issues, though, the variety of results we have presented, both qualitative and quantitative, demonstrate that the training wheels interface facilitated learning. Moreover, this facilitation is more than a mere release of error recovery time through error blocking. Nevertheless, this demonstration itself further raised a serious question of interpretation. On the one hand, the training wheels interface can be

thought of as a tactical technique for salvaging the learnability of difficult system interfaces. But on the other hand, it can be thought of as a strategic technique for architecting the function of systems and incorporating online training into system designs. These two views are not contradictory, but they are different.

On the first view, the training wheels interface is an illustration of a simple technique for retrofitting a user interface to achieve enhanced ease-of-learning. In our case, the training wheels alteration was made to an existent commercial system, at the assembly code level, in less than one man-month. Given the empirical results of our experiment, one could indeed hazard that we have succeeded in giving the learnability of a state-of-the-art system a little push in the right direction, and at a modest man-month cost. But is there any longer term implication in the training wheels interface?

Clearly, a more significant and longer term goal in system design is that of defining a user interface architecture with which to confront learnability and training issues, and with which to eliminate the need to retrofit for learnability in the first place. This is not a simple issue; even the first-order distinctions in user interface architecture are only now being thrashed out (e.g., [2]). Nevertheless, it is worth reflecting on the proposition that the training wheels approach, namely "staging" the presentation of function to users, is an architectural principle.

Consider the current conflict between the rapid diversification of user interface style and system function, on the one hand, and the goal of providing effective training for new users, on the other. Standard approaches to training, namely, online tutorials and self-instruction manuals, have failed up to now to resolve this conflict—and it is arguable that they are indeed losing ground as systems become more diverse and complex. A training wheels interface provides an alternative approach for the design of training: an initially simple and error-cushioned system interface to help develop a foundation of concepts and confidence, and a bridge from mastery of the basic function to acquaintance with the complete system (by keeping menus, messages, prompts and hardware identical between the training wheels interface and the complete system interface).

The rationale for such an approach is that a training wheels interface provides an exploratory environment [1] to the new user, an environment that affords active involvement in the learning process (learning by doing) with reasonable protection from the consequences of errors that active learning inevitably entrains [3, 4]. This is in contrast to standard approaches to the design of training (including online computer tutorials) which place the learner in a relatively passive role, and which either obstruct learner initiated activity (most online tutorials allow only one "correct" answer in any input field) or discourage it and provide no protection from the consequences of errors (most self-instruction manuals are written under the assumption that learners will follow them to the letter, and never make a mistake). Of course all bets are off if system interfaces should ever become so simple that learning is virtually instantaneous and training unnecessary. There is, unfortunately, nothing to indicate that any such circumstances are in the offing for the foreseeable future.

A new user is very much like an out-of-towner set down in the middle of a strange city. A goal, a map, and some directions all may help, but the possibilities for getting lost are overwhelming. Most insidious of all is the fact that if the out-of-towner makes a wrong turn he or she may not even become aware of it, because *everything looks strange.* The present research suggests that a practical alternative to the current methods of inducing new users into the mysteries of small systems is to create an environment in which the learner is either correct or else is corrected without penalty. The consequences of errors are blunted, and appropriate feedback is immediate. In such an approach, the new user is able to take control of the learning situation and of the system, to do something real and recognizable immediately, and to avoid the side tracks and error tangles that loom so large today.

Acknowledgments. We are grateful to David Boor, who designed the implementation for the Training Wheels Interface. This collaboration made it possible to iteratively design a training environment from an empirically derived inventory of user errors. We also thank Clayton Lewis, Ann Gruhn, Mary Beth Rosson, and Maureen Ruskie who commented on several drafts of this paper. Clayton Lewis suggested the term "training wheels."

REFERENCES

1. Carroll, J.M. The adventure of getting to know a computer. *Computer 15*, 11 (Nov. 1982), 49–58.
2. Carroll, J.M. Presentation and form in user-interface architecture. *BYTE 8*, 12 (Dec. 1983), 113–122.
3. Carroll, J.M., and Mack, R.L. Actively learning to use a word processor. In *Cognitive Aspects of Skilled Typewriting*, W.E. Cooper, ed. Springer-Verlag, New York, 1983.
4. Carroll, J.M., and Mack, R.L. Learning to use a word processor: By doing, by thinking, and by knowing. In *Human Factors in Computing Systems*, J.C. Thomas and M. Schneider, eds. ABLEX, Norwood, N.J., 1984.
5. Ferguson, G.A. *Statistical Analysis in Psychology and Education*. McGraw-Hill, New York, 1971.
6. Mack, R.L., Lewis, C.H., and Carroll, J.M. Learning to use word processors: Problems and prospects. *ACM Trans. Off. Info. Syst. 1*, 3 (July 1983), 254–271.

CR Categories and Subject Descriptors: H.1.2 [**Models and Principles**]: User/Machine Systems—*human factors, human information processing*; H.4.1 [**Information Systems Applications**]: Office Automation—*word processing*

General Terms: Human Factors

Additional Key Words and Phrases: user interface architecture, usability, ease of learning, human-computer interaction, human learning, training, education

Received 7/83; revised 2/84; accepted 3/84

Authors' Present Addresses: John M. Carroll, Computer Science Dept., IBM Thomas Watson Research Center, Yorktown Heights, NY 10598; Caroline Carrithers, Psychology Dept., Columbia University, New York, NY 10027.