In S. Doheny-Farina, (Ed.),
"Effective Documentation:
What We Have Learned From
Research." Cambridge, Mass:
MIT Press, pp. 47-72,
(1988).

Studies of Elaboration in Instructional Texts

Davida H. Charney
The Pennsylvania State University

Lynne M. Reder
Gail W. Wells
Carnegie-Mellon University

Technical writers who must produce a manual or some other instructional text are often caught between conflicting goals. On the one hand, as advocates of the readers who must understand and use the text, writers worry about leaving out any bit of information that might be important or useful. On the other hand, as employees who are accountable for producing a text at the least possible cost, they must use text sparingly: The longer the text, the higher the cost. In short, writers must constantly judge whether the importance of an explanation or some other piece of information is worth the cost of printing it. In some sense, all writers face the same fundamental question: what information should a text contain and to what extent should that information be elaborated?

In this chapter, we review several years of experimental research on this question of content and elaboration. Obviously, there can be no absolute answers to such questions. The answers must depend on factors such as the writer's purpose, the readers' intentions and abilities (i.e., their reasons for reading and their prior familiarity with the subject matter), and general human capacities for acquiring information or skills, in addition to conventional constraints on the form of the text. To narrow down the question to manageable proportions, we have focused our research primarily on individuals reading in order to learn a skill. Specifically, we focus on people who are learning to use a computer by reading a user manual.

The general strategy for our research has been to produce several versions of a computer manual that differ in systematic ways. We asked participants (generally college students) to work through the manuals. In some studies, we simply had participants read the manual; in others, participants also learned by working interactively with the computer. After removing the manuals, we asked the participants to demonstrate what they had learned by performing a set of tasks on the computer. We observed how many tasks the participants completed successfully and how long it took them to do so. By comparing the performance of participants who read the different versions of the manual, we can draw inferences about the characteristics of manuals that lead to better performance. We also employed a variety of readers in our studies: experienced computer users as well as computer novices, readers who opened a manual with a particular task in mind as well as readers who had no particular agenda. By comparing the performance of these groups of readers, we can draw inferences about the different needs of different readers.

Overview of the Chapter

This chapter is organized into four sections.

1. **Elaboration and Fact Learning**: We begin with a familiar instructional situation, reading in order to learn facts. We summarize a series of studies of how elaborations affect learning the main points in a standard college textbook.
2. **Elaboration and Skill Learning**: This section concerns the role of elaborations in skill learning. After contrasting two common viewpoints on elaborations in computer user manuals, we describe a series of studies in which we varied both types of elaborations and groups of readers.
3. **Learning by Reading, Watching, Discovering, and Working Exercises**: In this section, we consider the design of the tutorial section of user manuals, describing studies that call current designs of tutorials into serious question.
4. **Conclusion**: After summarizing our findings, we consider whether our findings may apply to texts other than computer manuals. We close by discussing how experimental research and user testing fit into the document design process.

ELABORATION AND FACT LEARNING

The question of how much elaboration to include in instructional texts was initially explored in a series of experiments with academic prose (Allwood, Wikstrom, & Reder, 1982; Reder, 1982; Reder & Anderson, 1980, 1982). The studies explored a seemingly simple question: Do elaborations help students master the central ideas in a standard introductory college textbook? To answer this question, Reder and her colleagues took chapters from several widely used college textbooks and prepared summaries of those chapters that were one-fifth as long. Students who participated in the studies either read the original chapter verbatim or read the summary. Then all participants were tested on their comprehension and recall of the main points.

Surprisingly, in study after study, the students who had read the summaries significantly outperformed the students who had read the full length chapters. The advantage for the summary group held up under a variety of circumstances. Some studies varied the timing of the test: whether the test occurred immediately after reading or after a long delay (up to one year). The type of test was varied: true/false, short answer, or free recall. Even the reading conditions were varied. In one study, students were allowed to take the materials home to read at their leisure; in other studies, the students' exposure to the materials was carefully timed and controlled. In some studies, the students read the text as connected prose; in other studies, the text was presented one line at a time. The pervasive finding was that students learned the main facts best when they studied them without elaboration.

The explanation of these rather counter-intuitive results seems to be a combination of two phenomena, one involving the ease with which information is registered or encoded in memory and the other involving the process of retrieving information from memory. Research on encoding has shown that the more time and attention a person spends studying a particular fact, the stronger that fact is encoded in memory (Bugelski, 1962; Cooper & Pantle, 1967). So whenever a student reads elaborations in a text, the elaborations divert her time and attention away from the main points that she must learn for the test. Conversely, when the student studies a summary, she can devote full attention to exactly those points that she must learn. This encoding phenomenon does not, however, completely account for the summary group's advantage. In one study, Reder and Anderson equated the total time that students in the summary group and students in the full chapter group spent studying the main points (Reder & Anderson, 1982). The students who read the full length chapters were given *extra* time to read the elaborations. Presumably, the encodings for the main points in both groups were equally strong, because students in both groups spent the same amount of time reading them. However, students in the summary group still performed significantly better on the tests. So, the handicap posed by elaborations seems also to involve retrieval: having the elaborations in memory seems to make it harder to retrieve the main point. In addition, the elaborations may make it harder for readers to distinguish important points from unimportant ones.

The finding that elaborations are often ineffectual and sometimes even detrimental to learning is a serious and curious charge. The implications for textbook production would be grave if one generalized this result to standard learning situations. One factor that we believe is crucial to the effectiveness of elaborations is the kind of learning that is expected to take place. Indeed, educators and laymen alike emphasize that it can be much less important to know a set of facts than to know how to use those facts. One goal of our research was to discover whether the advantage for summaries (or unelaborated text, generally) would remain when readers needed to *apply* the information to practice a skill, rather than simply *remembering* it. For example, reading full-length textbook chapters might be much more useful when the task is to write an essay, since composing requires deep understanding of the material and appropriate selection and development of various points.

In order to test the possibility that elaborations help people apply their knowledge, we turned our attention to computer user manuals since the task of learning to use a computer requires direct application of the information in the text.

ELABORATION AND SKILL LEARNING

Most people initially shared the intuition that elaboration is beneficial for learning facts; however, the value of elaboration for skill learning has been controversial for some time. For example, writers and researchers of computer documentation tend to fall into two quite distinct camps: the "expounders," who believe that instruction should be as complete and as explicit as possible, and the "minimalists," who believe that instruction should above all be brief and that it should leave much to the learner's own exploration.

The Case for the Expounders

The expounders' view is the more traditional: an instructional manual for novice learners should be as complete as possible; it should assume little if any prior knowledge and it should provide detailed exposition of all relevant points. Tausworthe (1979), for example, outlines several levels of detail for documenting computer software. The greatest degree of detail is called for in what he labels "Class A documentation," which he describes as follows:

> "Class A documentation is the most detailed; it contains specific definitions and detailed descriptions of every significant factor or item within the software specification..... This level of detail probably finds its most applicability in user manuals, and rightly so: The writer of a user manual is generally unavailable for consultation, so the user needs the extra detail." [Tausworthe (1979), pp. 158-159.]

For a more recent statement of a similar position, see Price (1984).

While the studies of fact learning described earlier suggested that elaboration does not always facilitate learning, we believed that elaborations might be much more beneficial for skill learning. Our first set of studies tested this possibility by comparing how well people learned skills from elaborated and unelaborated instructional texts.

STUDY 1. Evidence supporting the inclusion of elaboration emerged from our initial experimental study (reported in detail in Reder, Charney, & Morgan, 1986). For this study, we prepared two versions of a computer user manual that described the basic commands for IBM Personal Computer's Disk Operating System (PC-DOS). One version of the manual, the Elaborated version, contained many definitions, analogies, examples, overviews, explanations, and other elaborations. The Unelaborated version of the manual was about one-third as long (3500 words as opposed to 11,000), and omitted the elaborations. Appendices 1 and 2 present typical excerpts from the Elaborated and Unelaborated versions of the manual, respectively.

The participants in our study were 40 inexperienced computer users. We gave each participant one of the two manuals to read for forty-five minutes. Then we took away the manual and asked the participants to perform some tasks on the computer. (The subjects knew that the manual would not be available after the reading period.)

In addition to varying the amount of elaboration in the manuals, the experiment also simulated two common reading situations. Sometimes people have specific goals in mind and turn to instructional materials to find information relevant to those goals. At other times, people come to learn a new skill with only a general idea of how they will make use of what they learn. We simulated these two reading situations by dividing our participants into two equal groups. We gave one group advance information about the tasks they were going to perform before we gave them the manual to read. We assumed that participants in this "Task Orientation" group would then read the manual with the specific tasks in mind. The other participants (the "General Orientation" group) had no idea as they read the manual what kind of tasks we would ask them to perform. Within these two groups, half of the participants read the elaborated manual and half the unelaborated manual.

The tasks that the participants performed called directly on procedures described in the manuals: renaming files, creating subdirectories, copying and deleting files, and so on. As the participants worked at the computer, it recorded the commands and the time at which they were entered. We measured how well participants performed the tasks by counting how many tasks they were able to complete and how efficiently they worked (i.e., how much time they took and how many commands they had to issue to the computer).

Table 1: Mean Performance at Test as a Function of Elaboration and Participant Orientation, Study 1

	TASK ORIENTATION		GENERAL ORIENTATION	
	Elaborated Manual	Unelaborated Manual	Elaborated Manual	Unelaborated Manual
Proportion of Tasks Correctly Completed	.80	.80	.85	.76
Time to Complete All Tasks (in minutes)	33.5	36.1	29.4	40.2
Number of Commands Issued to Complete All Tasks	95.8	94.2	76.8	101.8

The results, summarized in Table 1, indicate that participants completed about the same number of tasks successfully with either manual. However, the results showed very different trends for how efficiently the Task Orientation and General Orientation groups completed those tasks. The Task Orientation group performed much more efficiently after reading the short, unelaborated version of the manual. On the other hand, the General Orientation group performed much better with the longer, elaborated manual. Table 1 shows this pattern most clearly for the average number of commands participants issued to complete the tasks. We infer from these results that both versions of the manual conveyed the basic information adequately, but that the elaborations influenced how efficiently readers could apply the information.

Even though we found that the shorter, unelaborated manual worked better for the Task Orientation group, the results in general support the expounders' case. Writers of instructional texts cannot assume that all learners will come to the manual with such clearly defined goals as the Task Orientation group. In fact, the participants who read the unelaborated manual without having specific tasks in mind consistently performed poorly. On balance, these learners were more greatly impeded by "under-elaborated" texts than the task-directed learners were by the "over-elaborated" version.

Study 1, then, provides some support for the traditional view that instruction should be complete and explicit. As commonly observed, however, there are disadvantages to complete and explicit documentation. In particular, users resist reading commercial manuals written according to the traditional guidelines, even when the relevant passages are easy to locate. People seem to prefer to figure things out on their own, or to ask someone for help (Wright, 1983; Scharer, 1984; Carroll, 1984). From this perspective, providing complete and detailed instruction seems to be of little practical use to the learner. Minimalist documentation grew out of an attempt to address this issue of reader motivation.

The Case for the Minimalists

Designers of so-called "minimalist training materials" proceed on the assumption that willingness to read a manual is inversely related to the manual's length, that people in general want to start doing things instead of reading about them and that therefore, instructional materials should actively encourage exploration by providing as little prose as possible. As Carroll (1984; 1988, reprinted in this volume) describes it: "The first principle of Minimalist design is to slash the verbiage; that's where the name comes from (i.e., less to read can mean better training)." Carroll, et al. put this principle into practice in a tutorial manual for a word processing system (the IBM Displaywriter System) and produced a revised manual that was one-fourth the length of the original.

Carroll, et al.'s (1988) procedure for shortening the manual included two major steps. First, they eliminated everything in the manual that was irrelevant to the task at hand (composing and printing a letter). Then, they took what was left, the relevant information, and deleted parts that readers should be able to learn on their own. It is worth emphasizing that Carroll, et al. were very selective about what information to omit. Their "missing information" is therefore quite different from the all too common blunder made in many commercial computer manuals of blithely or inadvertently leaving out crucial steps.

It is also important to note that Carroll, et al. (1988) made other changes to the minimal manual in addition to making it shorter: they clarified the terminology and organized the discussion around typical situations for users. This final category of changes undoubtedly improved the manual; however, it also obscures the results to some extent: we do not know how much the results are due to differences in elaboration and how much they are due to these other clarifications.

The manuals in Carroll, et al. (1988) were tutorial in the sense that readers were expected to try things out as they read about them. Carroll, et al. found that after working through the minimal version of the manual, participants (mostly secretaries) learned the same basic information more quickly than participants who used the commercially developed version of the manual. Furthermore, when participants went on to study more advanced topics, they learned new techniques more quickly if their initial training had been conducted with the minimalist materials. Carroll, et al. admit that these initial efforts at designing minimalist materials were exploratory (e.g., as a result of the testing, they put back some explanatory sections as well as some procedures that participants actually couldn't figure out on their own). However, their findings in the main support the minimalist position: having less to read led to equivalent or better learning at a faster overall rate.

Paradox Resolution: Identifying Components of Skill Learning

The preceding two sections leave us in a seemingly paradoxical position: The minimalists and the expounders both have experimental evidence to support them. The explanation for the different results may arise from differences in the prior experience of the participants. The participants in Carroll, et al.'s (1988) study were mainly secretaries who were very familiar with letter writing tasks. They may therefore have been more like the Task Orientation group in our study who also benefited from the less elaborated manual. An alternative explanation is that the tutorial aspect of Carroll, et al.'s manual made elaboration less necessary. We will reconsider the issue of tutorials later. For now, it is worth considering a third possibility: that depending on the situation, sometimes the minimalists are right and sometimes the expounders are right.

It is possible for both expounders and minimalists to be right if we shift the focus away from length per se. Length is not really the issue. The expounders' goal is not to write the longest manual possible; they in fact try to produce texts that are as concise and relevant as possible, given what topics must be covered. In an important respect, the minimalists try to do exactly the same thing. Obviously, the difference lies in defining what this relevant, essential information is to include.

To delve deeper into the issue of relevance, we began to consider what kinds of things people have to learn in order to acquire a skill (Charney & Reder, 1986; Reder, Charney, & Morgan, 1986). Perhaps some aspects of skill learning are easier when the instruction pertaining to them in the text is elaborated, while other aspects do not require elaborated instruction. If so, then the expounders may be including irrelevant information by giving detailed treatment to everything and not just the points that need it. Conversely, the minimalists may be underspecifying some points when learners would benefit greatly from more elaboration. In order to provide only relevant elaboration, then, it is necessary to isolate the aspects of skill learning and determine which ones may be facilitated by elaborated instruction. As we considered what is involved in learning a skill, we isolated three components. In order to perform a new skill well, a learner must:

1. Appreciate the meaning of novel concepts and the purpose of novel procedures. For example, proficient typists who have never used a word-processor must learn what things can and cannot be done on the computer. They must appreciate both the availability of automatic margin adjustment and the impossibility of underlining by overstriking what has already been typed.
2. Execute the procedures correctly. Learners must remember such details as where to position the cursor, in what order to type the arguments of a command, and whether a carriage return is required.
3. Use the procedures at the appropriate times. Learners must remember to use the procedures they have learned and know how to choose the most appropriate procedure for a particular situation.

Elaborations in a computer manual may touch on any of these topics: what concepts and procedures are involved, when they are relevant, and how one applies them. In Study 1, we used manuals that either elaborated on all of these points or on none of them.

Elaborating Different Types of Information

STUDY 2. In order to test the possibility that different components of skill learning require different degrees of elaboration, we classified the elaborations in the PC-DOS manual into two categories (Reder, Charney, & Morgan, 1986). Elaborations were classified as "conceptual" if they concerned basic concepts, such as the purpose of the Rename command, or conditions for application, such as when it is a good idea to rename a file. That is, conceptual elaborations dealt with both the first and third components of skill learning. Elaborations were classified as "procedural" if they concerned the second component, learning to issue commands correctly. Procedural elaborations included examples of correct commands, details about notational conventions, and so on. We then tested all possible combinations of the conceptual and procedural elaborations by producing four versions of the PC-DOS manual with the following combinations of elaborations:

- Rich Conceptual & Rich Procedural Elaborations
- Rich Conceptual & Sparse Procedural Elaborations
- Sparse Conceptual & Rich Procedural Elaborations
- Sparse Conceptual & Sparse Procedural Elaborations

The version that was rich in both types of elaboration was equivalent to the elaborated manual in our previous study (Appendix 1) and the version with neither type was equivalent to the completely unelaborated manual (Appendix 2). The other two versions are illustrated in Appendices 3 and 4, respectively. By using all four versions, we could determine whether the advantage we found for the elaborated manual in our first experiment was due to the conceptual elaborations, the procedural elaborations, or both.

This experiment was conducted very similarily to Study 1, except that no participants were given advance information about the tasks they would perform. This time, our participants included 40 novice computer users and 40 experienced computer users, none of whom had ever used an IBM-PC. We expected that the novices might need elaborations of both kinds, while the experienced computer users (who were already familiar with basic computer concepts) might only need the procedural elaborations.

Table 2 presents results showing how quickly subjects using each of the four manuals completed the tasks and how many commands they issued. The most efficient participants were those who had read the manuals containing rich procedural elaborations (see the two leftmost columns in Table 2). These participants finished the tasks in about 37 minutes and issued about 72 commands. In contrast, participants who read manuals with sparse procedural elaborations needed about 44 minutes and about 90 commands, significantly worse performance. Surprisingly, however, the conceptual elaborations seem to have had no effect at all. Adding rich conceptual elaborations to manuals with rich procedural elaborations was equivalent to having the procedural elaborations alone. Similarly, manuals with only rich conceptual elaborations produced performance no better than having no elaborations at all (see Table 2).

The results of Study 2 allow us to begin to sort out which components of skill learning require elaborated instruction and which do not. In particular, our participants needed little more than a summary of the conceptual information. They did not benefit from elaborations that described the concepts involved in using a computer (e.g., elaborate analogies that explained the concept of paths through subdirectories) or from advice about when and why particular commands were useful. What they did benefit from were well-chosen examples that illustrated what a correct computer command would look like in a specific plausible situation. For a more thorough discussion of these types of elaborations, see Charney and Reder (1987).

Table 2: Mean Performance at Test for Manuals with Four Combinations of Elaboration, Study 2

	RICH PROCEDURE		SPARSE PROCEDURE	
	Rich Concept	Sparse Concept	Rich Concept	Sparse Concept
Time to Complete All Tasks (in minutes)	37.4	37.7	43.5	45.9
Number of Commands Issued to Complete All Tasks	71.7	73.7	88.7	92.4

Other researchers have also found benefits for particular types of elaboration. For example, Kieras (1985) found that people learning to operate a mechanical device derive little benefit from detailed information about the internal workings of the device. However, he found that such "how it works" information is useful if the learner must *infer* the operating procedures (rather than memorizing them or looking them up) and if the "how it works" information is specific enough to enable such inferences.

The most surprising aspect of our results was the similarity of the patterns for novice and experienced computer users. Although the novices generally performed less well than the experienced computer users, they did not benefit any more than the experienced users from the conceptual elaborations. On the other hand, they did benefit just as much from the procedural elaborations. It is possible that the type of conceptual elaborations we included were not exactly what the novices needed. Nystrand's (1987) study is consistent with this possibility. Nystrand found that "high knowledge" and "low knowledge" computer users asked different kinds of questions when trying to use computer documentation. Low knowledge participants most often asked for "categorical definitions" while high knowledge participants most often asked for "further specifications." Nystrand characterized the differences in the questions as requests for topic elaborations (categorical definitions) or comment elaborations (further specification). He also found that adding the appropriate kinds of elaborations reduced the questions asked by both groups.

LEARNING BY READING, WATCHING, DISCOVERING, AND WORKING EXERCISES

While Study 2 does address some of the complaints of the minimalists, it does not address the central claim that people prefer *doing* things over reading manuals. Writers of commercial computer manuals have attempted to satisfy this preference by incorporating "tutorials" into user manuals. In this section, we will review evidence that, at least in their current form, such tutorials are inadequate for promoting skill learning. We will also discuss alternative forms of active learning that are more effective. In order to set this discussion in context, we will begin by briefly overviewing three active learning strategies that have received much attention in the cognitive and educational psychology literatures: learning by example, learning by discovery and learning by working exercises.

Learning by Example

Advocates of learning by example argue that texts should provide numerous worked-out examples that learners can use as models for solving problems on their own. A worked-out example consists of at least three parts: a specific problem to be solved, the correct solution to the problem, and an explicit sequence of steps that lead to the correct solution. The instructional text may also include the general rule that the problem exemplifies, as well as counterexamples. Worked-out examples can help people learn to recognize categories of problems and what solution strategies are appropriate to each category (Sweller & Cooper, 1985; Nitsch, 1977; Tennyson, Woolley, & Merrill, 1972). A worked-out example can also serve as a framework for constructing a solution to a new problem: the learner retrieves the example from memory and replaces terms specific to the old problem with terms relevant to the new one (e.g., Anderson, Farrell, & Sauers, 1984).

In some respects, Study 2 reported above supports the learning-by-example paradigm. Participants in Study 2 benefited from manuals that contained procedural elaborations, many of which consisted of commands that exemplified correct instantiations of an abstract syntactic "rule."

Features of the learning-by-examples approach may also be seen in the growing use of tutorials in computer manuals. The user of a tutorial follows step-by-step instructions to enter commands that demonstrate the computer's features. These tutorials are very similar to worked-out examples; the tutorial simply adds the physical activity of carrying out the instructions on the computer and allows the user to observe the computer's prompts and feedback messages at each stage along the way. Given the similarities between examples and the guided activity in tutorials, one might expect that people using tutorials would learn the same or more than people who simply read manuals that contain examples.

Discovery Learning

In complete contrast to the advocates of learning by example, advocates of discovery learning argue that people learn best when they set their own goals and explore a new domain with minimal guidance from a text or a teacher. Discovery learning maximizes the active involvement of the learner. Rather than studying problems set by a teacher (or the writer of the text), the learner invents his own problems, decides what techniques to use to try to solve the problem, and learns what works through trial and error.

Using a discovery learning approach, Carroll, Mack, Lewis, Grischkowsky, and Robertson (1985, reprinted in this volume) designed what they called a "guided exploration" manual for a word processing program. The manual described the basic features of the program very briefly but omitted certain procedural information in order to force learners to discover it on their own. Then they conducted a study comparing the Guided Exploration manual to a commercially produced tutorial manual. Participants using the guided exploration manual set their own "problems" (e.g., deciding to compose and print a letter) and tried out word processing procedures at their own initiative. Carroll, et al. found that this discovery learning group learned how to use the word processor in less time, finished criterion tasks more quickly and made fewer procedural errors than participants who worked through the tutorial manual.

Carroll et al.'s (1985) findings support our intuitions that active, hands-on involvement is superior to passively reading a text and examples, or even to obediently following tutorial exercises. However, we wonder to what other learning situations their findings will generalize. There are several reasons to believe that for some kinds of learners, discovery learning is not the optimal way to learn a skill, such as using a computer. In particular, discovery learning may cause certain kinds of problems for novices.

- Novice computer users may fail to create tasks for themselves that fully explore the system's capabilities. For example, a new computer user will probably not be able to invent a problem that clearly illustrates the uses of multiple windows or procedures for defining macros.

- Novices may never set themselves a task that demonstrates the advantages of one procedure over another. Unless they see examples of situations in which one procedure is more appropriate than another, novice computer users may stick with a procedure that they have already learned or that they find memorable even when it is significantly less efficient than some other procedure.

- Novices may develop and retain serious misconceptions unless their exploration leads to a highly salient error or problematic result.

Given these potential drawbacks to discovery learning, why did Carroll, et al. (1985) find advantages for the guided exploration manual over the tutorial manual? We conceive of discovery learning as involving two components. In the first component, learners decide to experiment with some procedure and invent a task involving those procedures. In the second component, learners work independently to solve the problems they have set for themselves. The advantages of discovery learning that Carroll, et al. found may have been due largely to this second component and not the first. However, the second component is closely related to another learning strategy, working exercises.

Working Exercises

The most familiar use of this learning strategy is in math and science textbooks in which exercises appear at the ends of chapters. Like worked-out examples, exercises set a specific problem. However, the answer to the problem and the steps for arriving at the answer are left for the learner to figure out (except when answers appear in the back of the book). Working exercises requires more active involvement on the part of the learner than studying worked-out examples; the learner must review the procedures described in the text, select appropriate procedures for a given problem, and determine how to apply the procedure to the specific case.

Working exercises differs from discovery learning in that the problems are set for the reader by the writer of the text. While these problems may not be intrinsically as interesting to the reader as those she might devise herself, they have some distinct advantages. First, the tasks may be designed to cover the full range of a system's capabilities. Second, tasks may be designed to illustrate situations in which one procedure is more appropriate than another. Finally, the tasks may be designed to anticipate and correct possible misconceptions.

It should be clear from the previous discussion that the three learning strategies lead to quite different prescriptions for what a computer user manual should look like. In our next study, we attempted to compare the effectiveness of manuals incorporating these strategies.

An Empirical Comparison of Learning Strategies

Study 3. In this experiment (Charney & Reder, 1986b), participants learned how to use the VisiCalc electronic spreadsheet. The study was conducted in two sessions: a training session and a test session that took place two days later.

In the training session, 30 participants (mainly undergraduates) studied 12 VisiCalc commands by reading a brief manual at their own pace. An example of an entry in the manual is provided in Appendix 5. Interspersed in the manual were training problem sets, consisting of three problems for each command. Participants were asked to solve the training problems whenever they appeared in the manual. While every participant read exactly the same text in the manual, the type of problem set for a given command varied from participant to participant. There were three types of problem sets:

- Tutorial. The instructions for the three problems in this set told participants exactly what keystrokes to type to reach the correct solutions. This problem set corresponded to tutorial activity.
- Exercise. The instructions in this set gave participants a specific goal for each of the three problems, but no guidance for how to reach it. Participants proceeded to solve the problems as best they could. After they finished or gave up, they were permitted to study our solution to the problem on the next page of the manual.
- Tutorial Plus Exercise. In this problem set, the first of the three training problems was a Tutorial problem and the remaining two were Problem Solving problems.

Appendix 6 presents a typical problem in its Tutorial and Exercise forms.

Every participant worked with all three types of problem set. That is, of the 12 VisiCalc commands that the manual discussed, four were presented with Tutorial training sets, four with Exercise sets and four with Tutorial Plus Exercise sets.

In order to compare learning with hands-on activity to learning by reading, we asked a separate group of 14 participants to read the manual without typing anything at the computer. The manual for this Read Only group presented the training problems in the same form as the Tutorial problems, except that participants were only allowed to study the steps, not enter them at the computer. In some sense, the training problems served the same function as worked-out examples in a textbook.

Two days after the training session, all participants were given a test consisting of 12 new problems to solve on the computer, one problem for each command. Their performance was evaluated in terms of their accuracy at solving the problems and the time they took to complete them.

Overall, as Table 3 indicates, training with exercises produced the most effective performance on the test. Participants were significantly more successful at solving test problems for commands that they had learned with Exercises or Tutorial Plus Exercises than for commands that they learned with Tutorial training alone. The form of the training problems did not affect how quickly participants could solve a problem. Table 3 also indicates that the computer interaction group performed better than the read-only group. These results support the notion that active learning situations in which people remember and apply procedures for themselves are more effective than situations in which people simply learn by studying the procedures.

Table 3: Mean Performance at Test as a Function of Types of Training Problems, Study 3

	COMPUTER INTERACTION GROUP			
	Tutorial	Exercise	Tutorial Plus Exercise	READ ONLY GROUP
Percent of Tasks Correctly Completed	.53	.66	.68	.48
Average Time Per Task (in secs.)	91	95	85	134

The overall picture that emerges from this study is inconsistent with the learning-by-example paradigm. Tutorials in general are similar to worked-out examples since both show the learner how to arrive at the correct solution. But neither the tutorial form of training nor the read-only training was very effective. More importantly, these two forms of training did not differ significantly: Carrying out the steps of the solution on the computer did not significantly improve learning over simply reading the examples in the manual. The results also indicate that our participants did not need to study an example before solving problems on their own. In particular, Tutorial Plus Exercises did not lead to better performance than pure Exercise training. While the results strongly support the efficacy of exercises, we would not conclude that exercises can completely replace examples, since our tasks required only simple applications of commands. However, the results do challenge the efficacy of existing tutorial manuals. Our evidence suggests that people learn no more from them than from simply reading the manual.

Study 3 did not directly compare exercises and discovery learning. However, our results suggest that the advantage which Carroll, et al. (1985) attribute to discovery learning may be due to the component whereby learners actively work out a method of completing the task they set for themselves. We conducted an additional study to see whether exercises are superior to discovery learning. Preliminary results suggest that learning was better when subjects were provided with exercises that set specific goals as compared with discovery learning (Reder, Charney, & Wells, in preparation).

The Placement of Exercises in a Manual

The exercises that we used in Study 3 were intended to help participants learn both *how* to apply procedures and *when* to apply them. The latter (the third component of skill learning described earlier) is one of the hardest things for an inexperienced computer user to learn. The knowledge that is needed for this component consists largely of knowledge of situations, or more precisely, associations of situations and procedures. This kind of situational knowledge is exactly what novices lack by definition: they have not *experienced* the range of situations that might arise and have not seen how to handle them effectively.

As mentioned previously, participants saw a set of three training problems for each VisiCalc command. In any of their forms (worked-out examples, tutorials, or exercises), the problems in a set illustrated a range of different situations in which the command might be useful. Yet simply applying a command three times in three different situations may not provide sufficient practice in learning to recognize when each command is most appropriate. We suspected that the timing of the opportunities to practice a procedure would have a great effect on how well people learn it. People learn better if their opportunities to study occurs at spaced intervals, rather than massing the study trials all at once (e.g., Glenberg, 1979; Madigan, 1969; Melton, 1967). We believed that this effect would generalize to skill learning because people need practice at recognizing the contexts of application. If training problems always follow the instructional text, our participants would not have to figure out which procedure was required, thereby missing practice on the third component of skill learning.

In Study 3, in addition to studying the effect of the form of training, we also investigated the effect of the placement or spacing of the training problems in the manual. For six of the twelve commands described in the manual, the training problems came all at once (the problems were "massed.") That is, immediately after reading about a given command, the participant saw three training problems that applied that command. For the other six commands, the training problems were "distributed." That is, participants saw only one training problem immediately after reading about the command. The other two problems appeared later, interspersed with problems that appeared after descriptions of other commands.

The results of the test revealed that participants were much faster at solving problems correctly for commands they had studied with distributed practice. The advantage for distributing the placement of the training problems appeared for all types of training problems (Tutorial, Exercise or Tutorial Plus Exercise). Overall, participants performed best when exercise form was combined with distributed placement.

SUMMARY AND CONCLUSIONS

We began this chapter with some surprising results that suggested that students learn the main ideas better from summaries of textbook chapters than from reading the chapters themselves. To explain these rather counter-intuitive results, we explored the possibility that elaborations are needed to promote application, rather than simply recall of the facts. Our findings on the most effective features of computer manuals address such factors as types of elaboration, types of readers, forms of interaction between the reader and the text, and methods of organization.

- With respect to *the degree and type of elaboration* in our manuals, we found:
    - No benefit from elaborations of general concepts (e.g., what is a disk drive);
    - No benefit from elaborations offering advice on when to apply specific procedures;
    - Significant benefits from elaborations on how to apply procedures (e.g., well-chosen situational examples).
- With respect to *computer users and their goals* for reading a manual, we found that:
    - Readers with specific tasks in mind need little or no elaboration in the text;
    - Readers without specific goals benefit from certain types of elaboration (i.e., those listed above);
    - Novice computer users and experienced computer users benefited from the same types of elaboration, although the experienced users were quicker and more accurate overall.
- With respect to the ways in which readers can *interact with the text*, we found that:
    - Readers who learned procedures simply by studying a manual containing worked-out examples performed most poorly;
    - Readers who learned procedures by carrying out the steps of a tutorial on the computer performed only slightly better;
    - Readers who learned procedures by working exercises that forced them to independently apply the information in the manual performed significantly better;
    - Readers who learned procedures by inventing their own problems to solve (discovery learning) may learn more than from a tutorial, but less than from working exercises.

With respect to the *organization of the manual*, we found that:

- Participants learned a procedure better when opportunities to practice it (either through tutorial or exercise training) were distributed throughout the manual, rather than only appearing immediately after the relevant instructional text.

THE RELATIONSHIP BETWEEN USER TESTING AND EXPERIMENTAL RESEARCH IN THE DOCUMENT DESIGN PROCESS

This chapter has concerned experimental research aimed at improving instructional texts through observations of readers attempting to learn from texts with various features. In concluding the chapter, we would like to comment on a related practice, user testing, and its relationship to the type of research we have reported.

The growing use of user testing is one of the most positive developments in technical writing in recent years. User testing is primarily a method of detecting and correcting problems in the draft of a document. To conduct a user test, writers give a draft of a document to a group of "users" (representatives of the intended audience of the document) and observe them using the document as they might on the job. For example, a group of consumers might be asked to read and carry out a set of instructions for assembling a stereo. The writers note places where the readers become confused or make mistakes. They use the results of user tests to revise the document which they then retest. They continue the cycle of testing and revision, producing successively better drafts of the document until it passes some criterion of acceptability. As a detection and correction mechanism, user testing represents the best means presently available to writers for ensuring that a specific document is complete, accurate and understandable from the standpoint of the intended audience. (For detailed discussions of methods of user testing, see Bond, 1985.) However, for all of its benefits, user testing is not sufficient for designing effective instructional texts. As we see it, user testing has three serious limitations.

(1) User testing simply cannot address the question of whether the writer has created the optimal document for conveying the desired information. Successive drafts in the user test cycle change largely in response to the results of earlier tests, rather than as a systematic exploration of alternative formulations. Further, user testing tends to focus on local rather than global features of the text. It is much more likely, for example, for user testing to reveal that a manual contains too much technical jargon than that it contains inappropriate elaboration. In essence, then, user testing is data-driven rather than theory-driven. That is, the decision to continue the cycle of testing and revision is governed by observations of problems in the draft rather than by theories or principles of document design. As a result, user testing provides few external guidelines for how good a document might ultimately become and little clue as to whether a radically different approach to presenting the text might not produce significant improvements in comprehensibility.

(2) It is difficult to extend or generalize from the results of a given user test. As discussed above, the methods chosen for altering the drafts of a document between tests are usually opportunistic rather than systematic. When the changes are unsystematic, it is impossible to determine which ones produced improvements in the test results. Another obstacle to generalization is that, for reasons of practicality, participants in any given user test are usually few in number and are selected from a highly specific population (i.e., the intended readers of the document). Because the sample of participants may be unrepresentative of readers in general, there is little assurance that revisions that are successful in one user test will also work for other groups of readers. This lack of generalization does not mean that user testing has no long term benefits. In fact, there is evidence that as writers conduct user tests, their sensitivity increases in detecting areas of text that will cause readers problems (Schriver, 1987). Our point here is that it is difficult to derive reliable principles for effective document design solely from the results of user testing.

(3) User testing enters the writing process at rather a late stage: after a draft has been produced. However, there are some things about readers that writers need to know much earlier, at the point at which they are generating ideas, selecting information and planning the overall shape of the document. At this early stage, writers are asking "What tends to work best for readers in this type of text?" as opposed to "Does our document work well enough for our readers?" Because user testing tends not to uncover general principles, it does not provide much help to writers at early points in the writing process.

Clearly, at an early stage in the document design process, writers need to draw on more general findings than user testing can provide, findings that only systematic experimentation can provide. The research we have reported here focuses primarily on the need for elaboration. Other experimental research (reviewed in Schriver, 1986; Felker, et al., 1980; and Wright, 1977) has aimed at making the information in a text easier to find (e.g., research on the use of headings, typography, and layout) and easier to comprehend and remember (e.g., research on vocabulary, sentence style, and order of presentation). While retaining a focus on documents and their readers, experimental document design research grows out of cognitive theories of how people learn from texts and what features of text facilitate reading and learning. Such research can provide valuable information to technical writers as they plan what information to include in their texts, how much to say about it, and how to present it. By seeking out such information in the early stages of the document design process and by user testing drafts at later stages, technical writers will maximize the likelihood that their readers will get the most out of their texts.

The work reported here was sponsored by the Office of Naval Research, Contract No. N00014-84-K-0063, and in part by Grant BNS-03711 from the National Science Foundation.

REFERENCES

Allwood, C. M., Wikstrom, T., & Reder, L. M. (1982). Effects of presentation format on reading retention: Superiority of summaries in free recall. *Poetics, 11*, 145-153.

Anderson, J.R., Farrell, R., & Sauers, R. (1984). Learning to program in LISP. *Cognitive Science, 8*, 87-129.

Bond, S. J. (June 1985). Protocol-aided revision: A tool for making documents usable. In F. Dwyer (Ed.), *Proceedings of the 1985 IBM Academic Information Systems University AEP Conference*. Alexandria, VA: IBM,

Bugelski, B. R. (1962). Presentation time, total time, and mediation in paired associate learning. *Journal of Experimental Psychology, 63*, 409-412.

Carroll, J., Mack, R., Lewis, C., Grischkowsky, N., & Robertson, S. (1985). Exploring exploring a word processor. *Human-Computer Interaction, 1*, 283-307.

Carroll, J., Smith-Kerker, P., Ford, J., & Mazur, S. (in press). The minimal manual. To appear in Human-Computer Interaction.

Carroll, J. (March 1984). *Designing MINIMALIST training materials* (Research Report 46643). IBM Watson Research Center, Computer Science Department. Also appeared in Datamation 30(18), 125-136, 1984.

Charney, D. H., Reder, L. M. (1986). Designing interactive tutorials for computer users: Effects of the form and spacing of practice on skill learning. *Human Computer Interaction, 2*, 297-317.

Charney, D. H., & Reder, L. M. (1987). Initial skill learning: An analysis of how elaborations facilitate the three components. In P. E. Morris (Ed.), *Modelling Cognition*. London: Wiley.

Cooper, E. H., & Pantle, A. J. (1967). The total-time hypothesis in verbal learning. *Psychological Bulletin, 68*, 221-234.

Felker, D. (April 1980). *Document design: A review of the relevant research* (Technical Report AIR-75002). Washington, D.C.: American Institute for Research.

Glenberg, A. (1979). Component-levels theory of the effects of spacing of repetitions on recall and recognition. *Memory and Cognition, 7*, 95-112.

Madigan, S. A. (1969). Intraserial repetition and coding processes in free recall. *Journal of Verbal Learning and Verbal Behavior, 8*, 828-835.

Melton, A. W. (1967). Repetition and retrieval from memory. *Science, 158*, 532.

Nitsch, K. E. (1977). Structuring decontextualized forms of knowledge. Unpublished doctoral dissertation, Vanderbilt University.

Price, J. (1984). *How to write a computer manual: A handbook of software documentation.* Menlo Park, CA: Benjamin/Cummings Publishing Co.

Reder, L. M. (1982). Elaborations: When do they help and when do they hurt? *Text, 2*, 211-224.

Reder, L. M., & Anderson, J. R. (1980). A comparison of texts and their summaries: Memorial consequences. *Journal of Verbal Learning and Verbal Behavior, 19*, 121-134.

Reder, L. M., & Anderson, J. R. (1982). Effects of spacing and embellishment on memory for the main points of a text. *Memory and Cognition, 10*, 97-102.

Reder, L. M., Charney, D. H., & Morgan, K. (1986). The role of elaborations in learning a skill from an instructional text. *Memory and Cognition, 14*, 64-78.

Reder, L. M., Charney, D. H., & Wells, G. W. (in preparation). Practicing the application of procedures in skill learning.

Scharer, L. (July 1983). User training: Less is more. *Datamation, 29*, 175-182.

Schriver, K., Hayes, J., Danley, C., Wulff, W., Davies, L., Cerroni, K., Graham, D., Flood, E., Bond, E. (April 1986). *Designing computer documentation: A review of the relevant literature* (Communication Design Center Technical Report No. 31). Carnegie-Mellon University.

Schriver, K. (1987). *Teaching writers to anticipate the readers' needs: Empirically-based instruction.* Doctoral dissertation, Carnegie-Mellon University,

Sweller, J., & Cooper, G. (1985). The use of worked examples as a substitute for problem solving in learning algebra. *Cognition and Instruction, 2*(1), 59-89.

Tausworthe, R. (1979). *Standardized development of computer software, Part II.* Englewood Cliffs, NJ: Prentice-Hall.

Tennyson, R., Woolley, F., & Merrill, M. (1972). Exemplar and nonexemplar variables which produce correct concept classification behavior and specified classification errors. *Journal of Experimental Psychology, 63*, 144-152.

Wright, P. (1977). Presenting technical information: A survey of research findings. *Instructional Science, 6*, 93-134.

Wright, P. (December 1983). Manual dexterity: A user-oriented approach to creating computer documentation. In A. Janda (Ed.), *Human factors in computing systems.* Boston, MA: Computer-Human Interaction Association,

APPENDIX 1

Excerpt of Elaborated PC-DOS Manual Used in Study 1

CHANGING THE CURRENT DIRECTORY -- CHDIR

The CHDIR command (short for "change directory") allows you to designate a directory as the "current" directory for a drive so that the computer will automatically look there for files or subdirectories mentioned in your commands. You can designate a current directory for each disk drive independently. Changing the current directory on the diskette in drive A does not affect the current directory on drive B. The root directory is automatically designated as the current directory for each drive when you first start up the computer. It is useful to designate a subdirectory as the current directory when you will be working primarily on the files in that subdirectory. Then you won't have to specify the path to the subdirectory in each command you issue.

FORMAT

The format of the command is:

```
CHDIR    [loc and name of new current directory]
```

You can use the abbreviation CD in the command instead of typing CHDIR.

[Location of new current directory] refers to the path to the directory you want to designate as the new current directory. The last directory name on the list should be the name of the directory you want to designate.

For example, the command below designates a subdirectory called PASCAL as the new current directory in drive B:

```
A> CHDIR  B:\PROGRAMS\PASCAL  <ENTER>
```

The first symbol in the path is a backslash (\). This means that the path to the new current directory starts with the root directory of the diskette in drive B. The path indicates that the root directory contains a subdirectory called PROGRAMS, and that PROGRAMS contains PASCAL, the directory you want to designate as the "new" current directory. As usual, the amount of location information you need to provide depends on which directory was last designated as the current directory for the drive.

To change the current directory back to the root directory, give a command like the following:

```
A> CHDIR  B:\  <ENTER>
```

The backslash (\) in the commands above symbolize the root directory. So the command above changes the current directory for drive B to the root directory.

If you forget which directory is the current directory, the computer can remind you. Enter a CHDIR command without specifying a location. The computer will display the path from the root directory to the current directory, or "\", if you are still in the root directory.

APPENDIX 2
Excerpt of Unelaborated PC-DOS Manual Used in Study 1

CHANGING THE CURRENT DIRECTORY -- CHDIR

The CHDIR command allows you to designate a directory as the "current" directory for a drive, so that the computer will automatically look there for files or subdirectories mentioned in your commands. You can designate a current directory for each disk drive independently.

FORMAT

```
CHDIR    [[d:]path]
```

You can use the abbreviation CD in the command instead of typing CHDIR.

If you designate a subdirectory as the new current directory, the computer will carry out all the subsequent commands within that directory, unless you specify a path to another directory. To change the current directory back to the root directory, use a backslash as the path.

If you forget which directory is the current directory, the computer can remind you. Enter a CHDIR command without specifying a location. The computer will display the path from the root directory to the current directory or a backslash if you are still in the root directory.

APPENDIX 3
Excerpt of PC-DOS Manual from Study 2 with RICH CONCEPTUAL and SPARSE PROCEDURAL Elaborations

CHANGING THE CURRENT DIRECTORY -- CHDIR

The CHDIR command (short for "change directory") allows you to designate a directory as the "current" directory for a drive so that the computer will automatically look there for files or subdirectories mentioned in your commands. You can designate a current directory for each disk drive independently. Changing the current directory on the diskette in drive A does not affect the current directory on drive B.

The root directory is automatically designated as the current directory for each drive when you first start up the computer. It is useful to designate a subdirectory as the current directory when you will be working primarily on the files in that subdirectory. Then you won't have to specify the path to the subdirectory in each command you issue.

FORMAT

The format of the command is:

```
CHDIR    [[d:]path]
```

You can use the abbreviation CD in the command instead of typing CHDIR.

If you designate a subdirectory as the new current directory, the computer will carry out all the subsequent commands within that directory, unless you specify a path to another directory. To change the current directory back to the root directory, use a backslash as the path.

If you forget which directory is the current directory, the computer can remind you. Enter a CHDIR command without specifying a location. The computer will display the path from the root directory to the current directory, or "\", if you are still in the root directory.

APPENDIX 4

Excerpt of PC-DOS Manual from Study 2 with SPARSE CONCEPTUAL and RICH PROCEDURAL Elaborations

CHANGING THE CURRENT DIRECTORY -- CHDIR

The CHDIR command allows you to designate a directory as the "current" directory for a drive, so that the computer will automatically look there for files or subdirectories mentioned in your commands. You can designate a current directory for each disk drive independently.

FORMAT

```
CHDIR    [loc and name of new current directory]
```

You can use the abbreviation CD in the command instead of typing CHDIR.

[Location of new current directory] refers to the path to the directory you want to designate as the new current directory. The last directory name on the list should be the name of the directory you want to designate.

For example, the command below designates a subdirectory called PASCAL as the new current directory in drive B:

```
A> CHDIR   B:\PROGRAMS\PASCAL   <ENTER>
```

The first symbol in the path is a backslash (\). This means that the path to the new current directory starts with the root directory of the diskette in drive B. The path indicates that the root directory contains a subdirectory called PROGRAMS, and that PROGRAMS contains PASCAL, the directory you want to designate as the "new" current directory. As usual, the amount of location information you need to provide depends on which directory was last designated as the current directory for the drive.

To change the current directory back to the root directory, give a command like the following:

```
A> CHDIR   B:\  <ENTER>
```

The backslash (\) in the commands above symbolize the root directory. So the command above changes the current directory for drive B to the root directory.

If you forget which directory is the current directory, the computer can remind you.

Enter a CHDIR command without specifying a location. The computer will display the path from the root directory to the current directory or a backslash if you are still in the root directory.

APPENDIX 5

Excerpt from VisiCalc Manual, Study 3

MOVE COLUMN OR ROW

The Move command moves the entire row or column that contains the current cell to another position on the worksheet.

PROCEDURES:

/M [FROM] . [TO] [RETURN]

Moves the contents of row or column in the [FROM] coordinate to the row or column specified in the [TO] coordinate.

The Move command requires the following information:

- The FROM Coordinates: The coordinates of a cell in the row or column that you wish to move. Visicalc automatically fills in the coordinates of the current cell (e.g., D5) as the FROM coordinates. If the current cell is not in the row or column you wish to move, type [BKSP] to erase these coordinates and type the coordinates of a cell in the row or column you want to move. Then type a period. Three periods appear on the edit line. Now you can type the "TO" coordinates.
- The TO Coordinates: The coordinates of a cell specifying the destination of the move. The TO coordinates must contain either the same column letter or the same row number as the FROM coordinates. The VisiCalc program determines whether to move a row or a column by comparing FROM and TO coordinates: if the column letter in the two coordinates is the same, then a row is moved; if the row number is the same, then a column is moved.

  The difference between the FROM and TO coordinates tells VisiCalc where to put the moved information. If the FROM coordinates (e.g., D5) have the same *column* letter as the TO coordinates (e.g., D3), then the contents of row 5 will move up to row 3. If the FROM coordinates (e.g., D5) have the same *row* number as the TO coordinates (e.g., B5), then the contents of column D will move left to column B.

VisiCalc makes room for the row or column you move by shifting the other rows and columns over. So moving a column or row to a new location does not "cover up" any other entries.

APPENDIX 6
A Typical Training Problem Presented in Tutorial and Exercise Forms, Study 3

A. TUTORIAL TRAINING

Alphabetize the names by putting the rows containing Steele and Stewart further down in the appropriate spots. Start with cell A1 as the current cell.

TYPE THIS:

```
/M . A7 [RETURN]
/M . A7 [RETURN]
```

B. EXERCISE TRAINING

Alphabetize the names by putting the rows containing Steele and Stewart further down in the appropriate spots.

Feedback, appearing on the following page:

You could have used the following sequence of commands (starting with cell A1 as the current cell) to solve the preceding problem.

```
/M . A7 [RETURN]
/M . A7 [RETURN]
```

C. CONTENTS OF VISICALC DISPLAY

```
   A          B
1 Steele     Clerk
2 Stewart    Clerk II
3 Sanders    Manager
4 Schiff     Manager
5 Sebert     Accountant
6 Snyder     Sec'y
7 Sweet      Clerk III
```