ELSEVIER

Learning and Instruction 16 (2006) 139–153

Learning and Instruction

www.elsevier.com/locate/learninstruc

Acquisition of procedures: The effects of example elaborations and active learning exercises

Richard Catrambone*, Mashiho Yuasa

School of Psychology, Georgia Institute of Technology, Atlanta, GA 30332-0170, USA

Abstract

This study explored the effects of active learning and types of elaboration on procedure acquisition (writing database queries). Training materials emphasized elaborations of conditions for executing actions versus elaborations of the connection between conditions and actions. In the ''active'' conditions, participants performed structured exercises designed to encourage active processing. In the ''passive'' conditions, participants studied examples that contained instructional elaborations. Although excessive instructional information for more knowledgeable learners can hurt performance, our results indicate that condition–action elaborations improved procedural performance the most, in both the active and passive conditions. Active learning required longer training time but was offset by reduced test time.

Keywords: Instruction design; Procedure learning; SQL

A person's active involvement in the learning process appears to produce the most robust and flexible learning (e.g., Chi, Bassok, Lewis, Reimann, & Glaser, 1989; Renkl, 1999). *Active learning* is defined here as students producing self-explanations while engaging instructional material. The best instruction and training programs for knowledge acquisition should therefore be those that induce or enhance some type of active information processing on the part of the learner (Dufresne, Gerace, Hardiman, & Mestre, 1992).

Merrill, Reiser, Merrill, and Landes (1995; see also Merrill, Reiser, Ranney, & Trafton, 1992) found that students learned LISP programming more effectively with an intelligent LISP programming tutor than with more traditional instructional learning. They attributed this success to hints and explanations given by the tutor that enabled the learners to correct their errors and understand *why* their corrections were successful rather than leading learners to a solution by telling them the answer. Merrill et al. (1995) suggested that the computer-based hints and explanations were ''procedurally incomplete'', providing just enough explanation, but not too much, so that the participants were encouraged to articulate their reasoning and to focus on problem-solving processes.

Active learning enhances declarative (fact) acquisition because the learner generates connections among to-be-learned items or between a to-be-learned item and other related facts already known to the learner (Anderson, 1983; Reder, Charney, & Morgan, 1986). This produces the so-called *generation effect* (Jacoby, 1978)—better memory for self-generated items—that enables the learner to succeed in improving the probability of recall of these

* Corresponding author.
E-mail address: rc7@prism.gatech.edu (R. Catrambone).

0959-4752/$ - see front matter © 2006 Elsevier Ltd. All rights reserved.
doi:10.1016/j.learninstruc.2006.02.002

materials. One question is whether such effects apply to procedural learning. Learning a procedure is defined here as acquiring knowledge needed to construct the steps for solving a problem; it is not just learning facts about the domain, but also how to use them to construct a solution.

Renkl (1999) has argued that self-explanations (i.e., explanations *generated* by a learner) can be more effective than instructional explanations (i.e., explanations *given* to the learner) because the former type, if sufficient scaffolding is present, takes better advantage of the learner's prior knowledge, is better-timed, and can be more memorable (see also Atkinson, Derry, Renkl, & Wortham, 2000). Carroll and Kay (1985), while studying the effectiveness of prompts on people learning to use a word processor, found that giving participants exact procedures to be followed during training (a passive learning condition) did not improve long-term acquisition and transfer of word processing skills. On some transfer tasks, participants in the control group, who had to learn procedures with less guidance and therefore presumably acquired skills more actively, performed faster and with greater accuracy than the groups given the procedures more directly during training.

Chi and her colleagues (Chi et al., 1989; Chi & VanLehn, 1991) found that active learning correlated positively with level of later skill acquisition. In these studies, however, active learning was a dependent variable, obtained through talk-aloud protocols with weaker and stronger students; therefore, the connection between active learning and problem-solving performance was not demonstrated clearly given that the learners differed on a variety of dimensions such as prior academic performance.

One goal of the present study was to compare instructional elaborations (i.e., elaborations provided in the training materials) with self-generated elaborations guided by scaffolding (Renkl, 1999). The work outlined above suggests that instructional elaborations and self-generated elaborations have potential advantages and disadvantages in terms of training time and subsequent problem-solving time and accuracy.

1. What types of elaborations are best for procedural learning?

The observations and findings described above highlight the importance of active learning for fact and procedure acquisition. However, they do not identify which aspects of elaborations are the most effective for procedure acquisition. Several studies provide some guidance on this issue.

Protocol studies by Chi and her colleagues (Chi et al., 1989; Chi & VanLehn, 1991) and Pirolli and Bielaczyc (1989) have indicated that while learning, a student might try to self-explain actions presented in text and examples or errors generated during his or her own problem-solving attempts. Chi and her colleagues suggested that explaining is a mechanism of study that allows students to infer and explicate the conditions and consequences of each procedural item in the example as well as to apply the principles and definitions of concepts to justify them. That is, while studying examples, good students identify a context or situation in which a given action is appropriate and link relevant actions to that situation.

Pirolli and Bielaczyc (1989) also examined explanations generated by students while studying worked-out examples of LISP programming and created a classification of self-generated explanations. For present purposes, the two most relevant explanation classes concerned statements about the domain itself, LISP programming. Pirolli and Bielaczyc suggested that domain explanations can be divided into two types: (1) syntax-oriented explanations, referring to the syntax of program code and other surface features of the example; and (2) semantic explanations, involving an abstract interpretation of the process generated by a piece of the code.

The work cited above suggests that two types of elaborations are crucial for procedural learning. The first is the conditions under which a procedure or sub-procedure should be performed. This is important because learners often fail to understand *when* a procedure should be carried out. The second is the relationship between the goal and the necessary actions, that is, knowing *what* to do.

The second goal of the present study was to compare the effectiveness of the two types of elaborations described above. Initially, it might seem straightforward to provide (or lead learners to generate) elaborations on both conditions *and* actions in order to produce superior subsequent task performance. However, there are at least two reasons why this might not be the case. First, excessive information can produce too much cognitive load and interfere with schema development (Sweller, 1988; Sweller, van Merriënboer, & Paas, 1998), particularly if the additional information is not needed by that learner—the so-called "expertise reversal" effect (Kalyuga, Chandler, & Sweller, 1998). That is, sophisticated learners—which describes the learners in the present study—can actually suffer in their subsequent

task performance if the training materials contain information that could be inferred reasonably easily by the learner from other material in the lesson (referred to by Mayer, 2001 as the ''coherence principle'').

A second reason why too much elaboration could lead to decreased task performance is based on the findings of a program of research on minimalist documentation (Carroll, 1998; Carroll, Smith-Kerker, Ford, & Mazur-Rimetz, 1988; Catrambone & Carroll, 1987). These studies have indicated that learners can become lost in the details of instructions and that reduced documentation can produce superior transfer performance in a variety of situations. While the situations in which reduced documentation is superior have not been fully explicated, one guideline is that if the learner can be expected to be able to figure out certain details based on prior knowledge and/or information presented in the instructions, then he or she will perform better than when the documentation is more complete. Thus, we chose to contrast these two elaboration approaches: conditions versus conditions *and* actions.

The materials used in the present study contained all the ''facts'' learners would need to carry out subsequent tasks. Thus, one might expect high performance on a test of declarative knowledge by strong learners. However, the experimental manipulations were expected to impact learners' organization and depth of analysis of the information so that there might be differences in learners' success in constructing the steps for solving new problems. Thus, differences in procedural performance could be found despite an expected lack of difference in declarative knowledge.

It is worth noting that one might expect a possible interaction between degree of elaboration and the degree of active learning. For instance, briefer instructional materials paired with active learning exercises might be more likely to lead a learner to explicitly generate needed details relative to the same materials without such exercises. Conversely, more detailed materials with active exercises could produce too much cognitive load (Sweller, 1988) and therefore retard learning relative to the same materials without such exercises.

2. The test domain: the Structured Query Language (SQL)

SQL, the Structured Query Language for databases (sometimes referred to as the ''Standard Query Language''), is a command language for relational databases. It was chosen here as the test domain because writing a query with SQL is a relatively complex task and because the knowledge required to write queries can be fully specified (Smelcer, 1989).

A brief description of SQL is necessary before describing the experiment. A relational database of SQL consists of data organized into tables. Each table consists of columns and rows. The columns are labelled with column names (e.g., ID, HOMESTATE, MAJOR) and each row contains a set of values, one for each column. See Appendix A for an example of several tables in an SQL database.

Retrieving data from a database is the most common SQL operation. A database retrieval is called a query. Most SQL queries are composed of three clauses: SELECT, FROM, and WHERE. The SELECT clause specifies which columns to print (e.g., SELECT NAME, COURSE). The FROM clause specifies the tables in which those columns are found (e.g., FROM FACULTY, TEACHING). The WHERE clause specifies constraints on which items in the columns are to be printed (e.g., WHERE HOUR $>$ 3). If a query requires data from multiple tables, then a join condition is added to the WHERE clause (e.g., AND FACULTY.ID = TEACHING.ID).

For example, suppose one needs to find out the names of all instructors (NAME), and the courses (COURSE) they teach that have credit hours (HOUR) greater than 3 h. Further, suppose NAME is stored in one table (FACULTY) and COURSE and HOUR are stored in another table (TEACHING). If the FACULTY and TEACHING tables share a common column, ID (by ''share'' we mean that both tables have a column named ''ID''), then the correct query is

```
SELECT    NAME, COURSE
FROM      FACULTY, TEACHING
WHERE     HOUR > 3
AND       FACULTY.ID = TEACHING.ID;
```

3. Overview of experiment

The experiment was conducted in three phases. First, a participant read an abbreviated SQL manual. If the participant was in an active learning condition, structured exercises were given with the manual. Second, after studying the

manual, the participant was given a criterion test consisting of factual questions. When the participant could answer all questions correctly, he or she was allowed to proceed to the third phase, writing SQL queries.

There were four groups in the experiment that differed in terms of the type of information emphasized during training and whether the training involved active or passive learning. The names of the groups were the Condition–Action Pairing Active Group (CA-ACT), the Condition Elaboration Active Group (C-ACT), the Condition–Action Pairing Passive Group (CA-PAS), and the Condition Elaboration Passive Group (C-PAS).

The condition elaboration groups (C-ACT and C-PAS) received or generated information that elaborated the condition parts of condition–action pairs in their manuals. The C-ACT group received structured exercises that focused on elaborating the conditions (see Appendix B), whereas the C-PAS group received such information as an explanation for the example query (see Appendix C). When the exercise problem in the C-ACT lesson was answered correctly, the resultant statement was identical to the C-PAS explanation.

The condition–action pairing groups (CA-ACT and CA-PAS) received or generated information that elaborated the associations between the conditions and actions of query-writing procedures. The CA-ACT group received structured exercises stressing the associations between conditions and actions of query writing (see Appendix D), whereas the CA-PAS group received such information as an explanation for the example query (see Appendix E). When the exercise problem in the CA-ACT lesson was answered correctly, the resultant statement was identical to the CA-PAS explanation.

4. Hypotheses

4.1. Training time

Participants in the active conditions were expected to take longer to finish the training phase because they would be spending time answering questions. Of importance is whether this increased time will account for any performance differences found in the test phase. No particular prediction was made as a function of type of elaboration (condition versus condition–action) or the interaction with the active/passive manipulation.

4.2. Relationship of criterion performance to test performance

No relationship was predicted between performance on the criterion test and transfer test performance. The criterion test was administered in order to make sure that the participants knew the SQL operators and procedures; the rote learning of these operators and procedures was sufficient for successful performance on the criterion test. However, knowing the answers to the questions in the criterion test would be necessary, but not sufficient, for completing the transfer tasks successfully. Bovair and Kieras (1991) have argued that successful fact learning as measured by declarative memory tasks does not guarantee successful procedure execution. Successful performance on the transfer tasks in the present study was expected to depend largely on participants' knowledge of how to use the operators. That is, knowing that the operators would be useful for completing the test tasks only if the participants could identify the conditions and goals of the problem, retrieve appropriate operators for a given condition or goal, organize operators in a syntactically correct manner, and assign appropriate values for all the variable parts of the query.

4.3. Test time

Participants in the active conditions were predicted to solve the test problems more quickly because the practice they received during training of generating queries should make them faster at accessing relevant information (assuming the information is accurately represented in memory). Participants in the condition–action pairing conditions might be faster during the test because they might have stronger connections between goals and their associated actions.

4.4. Errors

As a result of careful examination of the type of errors typically made in a pilot study and in other research (e.g., Mitrovic, 2003; Welty, 1985), four error categories were identified. Table 1 presents a list of the most common errors

Table 1
Common errors during SQL query writing (grouped by error type)

Trivial errors
Misspellings
Syntax errors
Incorrect format in a join condition
Missing a semicolon at the end of a query
Incorrect logical operators in lieu of a pattern matching operator LIKE
Incorrect mathematical operators in lieu of a range operator BETWEEN
Missing quotation marks for character variables
Extra quotation marks for numeric variables
Incorrect variable position indicators used in a pattern matching condition
Imperfect knowledge and understanding of conditions, actions, and/or consequences of query
Omitting a join condition
Omitting a FROM clause
Omitting a search condition
Omitting one of multiple tables in a FROM clause
Substituting AND for OR and vice versa
Incorrect column names in a SELECT clause
Incorrect table names in a FROM clause
Non-optimal actions
Listing all columns in a table in lieu of an asterisk (*) as shorthand

grouped by error category. The first category was trivial errors such as misspelling column names. The second category was syntax errors such as missing or extra quotation marks. The third error type consisted of errors resulting from participants' imperfect knowledge and understanding of conditions, actions, or consequences of query writing. When participants made errors in this category, they were either unable to identify conditions presented in the problem statements accurately, or unable to produce proper actions associated with the conditions. The fourth error type was non-optimal actions such as listing all columns in a table in a SELECT clause instead of using an asterisk (*) as shorthand. Even though errors of this class were defined as non-optimal, the resultant display outputs would be correct. Of primary interest in the present study were syntax errors and errors due to imperfect knowledge.

Participants in the active conditions were predicted to make fewer syntax errors and errors due to imperfect knowledge than those in the passive conditions because the former groups had to generate parts of queries during training in a scaffolded framework; the structured exercises during learning should help them organize the procedures in a way suitable for performing these tasks. CA-ACT participants were predicted to make the fewest errors of all the groups because they generated both conditions and actions during training. CA-PAS participants might also perform relatively well given that they were exposed to elaborations for both conditions and actions; their performance relative to C-ACT will presumably be a function of the importance of self-generation versus the nature of elaborations.

No systematic difference was predicted among the groups in the frequency of trivial errors and non-optimal actions because the training manuals did not contain differential information specifically tailored to these issues.

5. Method

5.1. Participants

Sixty-four undergraduates enrolled in introductory psychology courses at Georgia Institute of Technology were recruited for this experiment and received course credit. Participants were not computer novices; however, those with previous experience with SQL or any other similar database languages were screened from the experiment. Computer-science majors were also excluded from this experiment.

5.2. Materials and procedure

The experiment consisted of three phases: (1) training, (2) criterion test, and (3) computer-based query-writing tasks. Each participant was tested individually and the experiment was self-paced.

Participants were assigned evenly and randomly to one of the four training conditions: Condition Elaboration Active Group (C-ACT), Condition Elaboration Passive Group (C-PAS), Condition–Action Pairing Active Group (CA-ACT), and the Condition–Action Pairing Passive Group (CA-PAS).

Participants learned about SQL query writing from a manual prepared by rewriting the *Introduction to SQL* (1987). The rewritten manual consisted of 11 lessons. Participants were provided with a stopwatch and asked to read the SQL manual carefully for at least 15 min but not longer than 25 min. By having some control over their own training, participants were expected to be able to allocate their study more naturally.

A reference sheet containing the example database used in the examples and structured exercises was provided to the participants and was available during training (see Appendix F). Each lesson began with a brief description of one SQL concept, operator, or function. The text was followed by an example query illustrating that concept, operator or function, and a resultant display output.

In the active conditions one structured exercise was presented at the end of each lesson, after the example. The exercises were designed to encourage participants to relate the example to the preceding text. The participants were able to find the answer to each exercise on the page following the lesson. After the participant answered the problem, he or she would turn to the next page and study the answer to the problem. The experimenter watched the participants as they worked in order to ensure that this rule was followed.

The structured exercises for participants in the C-ACT condition summarized and elaborated the condition in which the example query was appropriate. The structured exercises for participants in the CA-ACT condition focused on stating the conditions and actions of each SQL clause in the query in an ''IF–THEN'' form. C-PAS and CA-PAS participants studied lessons containing examples that started with a summary statement that was the answer participants tried to produce in the comparable C-ACT or CA-ACT lesson.

When participants finished the study phase, they were given the declarative knowledge criterion test consisting of 13 statements containing 20 blanks to be filled in. Each statement was taken from the text of the SQL manual, and some key words or phrases were replaced with blanks. All 11 lessons were covered in the test. Participants were asked to fill in the blanks with appropriate words or phrases in order to complete the statements. The participant was informed which items he or she had answered incorrectly (if any) and was told to correct them by referring to the manual. If the answers were still incorrect, the experimenter provided correct answers. The participant was then allowed to proceed to the transfer phase.

In the transfer phase, participants performed 11 SQL query-writing tasks (see Table 2). The tasks were presented in a fixed order and varied in difficulty. The experimenter provided a verbal and written description of the database participants would work with (see Appendix A for a subset of the database). Participants were instructed to study the database so that they could find information needed to perform the query-writing tasks comfortably. They were allowed to refer to this description while writing queries but they could not look at the training materials.

The query-writing tasks were presented on the computer screen one at a time via a program that approximated a typical SQL interface. However, there were two important differences between this program and a genuine SQL interface. First, if participants issued an incorrect query, the program displayed a feedback message stating that the query was incorrect and that they must redo the task. That is, participants were not allowed to proceed to the next query-writing task until they succeeded issuing a correct query. If a participant could not issue a correct query after three trials, he or

Table 2
Query-writing transfer tasks

1. Retrieve *all* available information about *all* students from table STUDENT.
2. Retrieve *all* available information about *all* faculty members from table FACULTY.
3. Print out the names and majors of *all* students.
4. Print out the course names and sections of *all* courses.
5. Print out the names and majors of all students who are Georgia (GA) residents.
6. Print out the names of *all* seniors.
7. Print out the names of all seniors with honors standing.
8. Print out the names and sections of all the courses whose size is greater than 50 or greater than its upper limit.
9. Print out the names of all faculty members and the courses and sections they are currently teaching.
10. Print out the titles of all courses and the ID numbers of the faculty members who are teaching them.
11. Print out the names and majors of all students on academic probation and the courses and sections taken by these students.

she was given a brief hint (see below). Second, all SQL syntax error messages were suppressed. This was done because SQL messages are often cryptic and because their helpfulness varies widely from one type of error to another.

Participants were informed that keystrokes and timing would be recorded and that they could spend as much time as necessary on each task but should not waste time. They were instructed to ask the experimenter for hints if they were unable to issue a correct query after three trials. The hints were prepared beforehand by examining query-writing error data and self-monitoring statements collected in the pilot study. A hint was a brief message of one- or two-sentence length. Appendix G presents a selection of prepared hints for different errors. A participant was first given a general hint that pointed out a clause where an error was located and described the general nature of the error (H1; Appendix G). If the participant still had difficulty, the second hint described the exact location of the error (H2). If the participant was still unable to correct the mistake, the third hint told him or her exactly how it should be corrected (H3). When participants made unanticipated errors (this rarely happened), the experimenter made up, on the spot, brief hints of one to two sentences that did not give away the solutions. Hints were given only one at a time even when a participant had more than one error in his or her query.

6. Results

6.1. Training: time spent studying the manual

Participants spent an average of 19.4 min studying the manual (see Table 3). Participants in the active conditions (21.85 min) took 29% longer than those in the passive conditions (16.95 min), $F(1, 60) = 17.78$, MSE = 21.36, $p < .001$, $f = .48$. There was no effect due to type of elaboration, $F(1, 60) < 1$, nor was there any interaction, $F(1, 60) = 2.86$, $p =$ ns.

6.2. Criterion test performance

Criterion declarative test performance was quite good: CA-ACT (average of 19.1 out of 20), C-ACT (18.2), CA-PAS (18.2), and C-PAS (18.1). These averages are for number of items successfully answered on the first try; recall that participants had to re-answer any items on which they were initially incorrect. There was no effect on first-try criterion test performance as a function of activity (active versus passive), $F(1, 60) = 1.97$, MSE = 2.03, $p =$ ns, type of elaboration, $F(1, 60) = 1.97$, $p =$ ns, or their interaction, $F(1, 60) < 1$.

6.3. Time to do transfer tasks

Participants spent an average of almost 25 min completing the transfer tasks (see Table 3). Participants in the passive conditions (28.70 min) were 38% slower than those in the active conditions (20.85 min), $F(1, 60) = 16.62$, MSE = 58.70, $p < .001$, $f = .43$. Participants in the condition elaboration conditions (29.94 min) were 29% slower than those in the condition–action pairing conditions (21.60 min), $F(1, 60) = 10.96$, $p = .002$, $f = .35$. There was no interaction, $F(1, 60) < 1$.

Given that there were differences among the groups in training time, the test time analysis was redone using training time as a covariate. The pattern of results did not change, thus suggesting that the superior performance of the

Table 3
Training and transfer time as a function of training condition

	Active learning		Passive learning		AVG
	Condition/action elaboration CA-ACT	Condition elaboration C-ACT	Condition/action elaboration CA-PAS	Condition elaboration C-PAS	
Training time (min)	21.8 (3.75)	21.9 (4.21)	16.5 (5.22)	17.4 (5.14)	19.4
Transfer time (min)	17.3 (6.94)	24.4 (6.54)	25.9 (5.9)	31.5 (10.5)	24.8
Total	39.2 (8.94)	46.3 (9.52)	42.4 (9.80)	48.9 (11.98)	44.2

Note: Standard deviations in parentheses.

active groups was not merely due to spending a longer time studying but rather was due to their greater understanding and organization of their procedural knowledge. As the error analysis below indicates, this understanding involves fundamental knowledge about query writing and does not simply reflect greater care in the trivial details of query construction. The superior performance of the condition–action groups compared to the condition elaboration groups suggests that providing information stating the connection between goals and actions is useful for procedural learning (Catrambone, 1998).

The test time analysis was redone using the number of hints as a covariate (the hint analysis is presented below). The rationale for using number of hints as a covariate is that the time needed to supply hints could have affected transfer time. However, when number of hints was used as a covariate, the pattern of results did not change.

6.4. Total time (training + test)

Participants spent an average of about 44 min completing the training and transfer tasks (see Table 3). Participants in the condition elaboration conditions (47.60 min) were 17% slower than those in the condition–action pairing conditions (40.80 min), $F(1, 60) = 7.28$, MSE = 102.49, $p = .009$, $f = .32$. There was no effect as a function of activity (active versus passive), $F(1, 60) = 1.35$, $p =$ ns. There was no interaction, $F(1, 60) < 1$.

6.5. Errors made when doing transfer tasks

The number of unimportant errors (trivial errors and non-optimal actions) and the number of important errors (syntax and imperfect knowledge) are shown in Table 4. Participants made an average of .62 *unimportant* errors when doing the transfer tasks. There was no effect of activity, $F(1, 60) < 1$, or elaboration, $F(1, 60) = 1.19$, $p =$ ns. The interaction was also not significant, $F(1, 60) < 1$.

Participants made an average of almost four *important* errors when doing the transfer tasks. There was no effect of activity, $F(1, 60) = 2.45$, $p =$ ns, but there was an effect of elaboration (CA: 1.85, C: 5.90), $F(1, 60) = 33.44$, MSE = 7.60, $p < .001$, $f = .59$. The interaction was not significant, $F(1, 60) < 1$.

If the analysis is repeated using training time as a covariate, the overall pattern of results does not change.

6.6. Number of hints required during transfer tasks

Participants required an average of 1.8 hints when they worked on the transfer tasks: CA-ACT (.50), C-ACT (1.47), CA-PAS (1.62), and C-PAS (3.69). There was an effect of activity (active: .98, passive: 2.66), $F(1, 60) = 10.51$, MSE = 4.26, $p = .002$, $f = .36$, and type of elaboration (CA: 1.06, C: 2.58), $F(1, 60) = 8.63$, $p = .005$, $f = .33$. There was no interaction, $F(1, 60) = 1.12$, $p =$ ns.

If the analysis is repeated using training time as a covariate, the overall pattern of results does not change.

Table 4
Number of errors on transfer tasks as a function of training condition

Error type	Active learning		Passive learning		AVG
	Condition/action elaboration CA-ACT	Condition elaboration C-ACT	Condition/action elaboration CA-PAS	Condition elaboration C-PAS	
Important errors (syntax and imperfect knowledge)	1.5 (1.41)	5.2 (3.42)	2.2 (2.07)	6.6 (3.52)	3.9
Unimportant errors (trivial and non-optimal)	0.38 (0.59)	0.75 (1.24)	0.62 (0.88)	0.74 (0.73)	0.62

Note: Standard deviations in parentheses.

7. Discussion

Active learning requires *more training time* but *reduces time on subsequent performance* compared to instructional elaborations. The longer training time appears to be recouped when one is carrying out new tasks, at least within the time frame of the current experiment. A longer time frame might allow performance differences to grow; alternatively, differences might become even smaller if experience begins to play a larger role on performance relative to initial training.

Materials designed to help learners focus on actions and the conditions for carrying out those actions reduced subsequent performance time as well as the number of *important errors* compared to materials that elaborated on the conditions only. Condition–action materials also reduced the number of hints learners needed to carry out new tasks. It appears to have been useful to elaborate both the condition and action parts of procedures and that this information was not overwhelming.

These findings concerning condition–action elaboration are of particular interest. At one level it seems obvious that successful procedure execution depends on accurate identification of conditions or goals *and* retrieval of appropriate actions that match the conditions or achieve the goals. The training materials for the condition elaboration and condition–action pairing groups contained this information, but the latter group had greater emphasis on the connection. However, a number of studies though have suggested that redundancy of information for good learners can lead to performance decrements (Kalyuga et al., 1998; Mayer, 2001; Sweller, 1988; Sweller et al., 1998); that is, if learners can be expected to reasonably infer some information, then it is better to promote this process rather than spelling out all details. The participants in the current study were bright, sophisticated, technically oriented students who, while unfamiliar with SQL, were experienced with computers and were generally good problem solvers.

Nevertheless, the results suggest that an emphasis on conditions *and* their associated actions lead to acquisition of skills needed to identify conditions described in the problem statements and to break them down to sets of subconditions. These subconditions have an appropriate ''grain size'' and can be matched relatively easily to actions of meaningful units. For example, in the present study, each condition–action elaboration exercise or explanation consisted of statements that described clause-level conditions and matching clause-level actions. On the other hand, condition-only elaboration exercises and explanations described conditions at a query level. Presumably, for novice query writers, elaboration that describes procedures for writing SQL clauses one at a time is more meaningful or useful than elaboration that describes a procedure for writing a single SQL query all at once. This would be consistent with other studies that have shown that instruction that focuses on smaller building blocks of procedures, a ''modular'' approach, produces better transfer to novel problems compared to ''molar'' instruction that attempts to teach novices to categorize problems and then apply category-appropriate procedures (Catrambone, 1998; Gerjets, Scheiter, & Catrambone, 2004). The latter approach makes it hard for learners to figure out how to adapt procedures while the former approach makes this process easier. In addition, a modular approach requires the learner to hold fewer items simultaneously in working memory during training, thus reducing cognitive load and speeding up the learning process (Sweller et al., 1998).

While most participants in the present study had little difficulty doing well on the criterion test, there were group performance differences on the query-writing tasks. This suggests that declarative knowledge measures are not necessarily a good predictor of successful procedure execution, at least when the declarative knowledge is over some threshold. As Bovair and Kieras (1991) suggest, procedure comprehension that leads to successful execution of the procedures requires different processing of the materials than successful declarative text comprehension demands.

8. Future work

Prior research has suggested that instructional information should not be unnecessarily detailed, particularly when a learner is likely to have the ability and background to infer details (Kalyuga et al., 1998; Mayer, 2001). Some studies have indicated that providing learners with a subgoal framework for solving problems, without specifying all the possible steps that might be needed for future problems (i.e., just showing steps for a particular problem), produces good transfer performance (Atkinson, Catrambone, & Merrill, 2003; Catrambone, 1996, 1998; Gerjets et al., 2004). Clearly this interacts with learners' prior knowledge given that some learners will need more guidance. Thus, the issue of what constitutes an appropriate level of detail remains an open one and will be determined by the grain size of the instructions, the difficulty of the domain, the learner's background, and probably other factors as well.

We wish to test the generalisability of the results obtained in this study with a less technologically sophisticated population of computer users. The majority of the participants majored in engineering or science and virtually all participants were quite comfortable with computers and computer jargon. Therefore, the participants in this study were able to read the training manual in a relatively short period of time without any assistance and to attain a fairly high level of performance. They might also have greater working memory capacity, and thus have been less susceptible to cognitive load limitations, compared to the general population. Will the sorts of elaborations used here, work for a more diverse population of computer users? Perhaps the expertise-reversal effect (Kalyuga et al., 1998) will be demonstrated such that less-sophisticated learners will do better with more elaborated materials while more-sophisticated learners will do better with less elaborated materials (with the materials used in the present study falling into the latter category). Will this augmentation make the documentation unwieldy, increase cognitive load, and reduce the motivation of users to read it? These are important practical questions.

Another extension to this work concerns differences between the study behaviour the participants in this experiment were asked to follow and the study behaviour of people who are trying to learn a new computer software package in a more natural learning situation. For example, computer users usually do not read a manual thoroughly, from the beginning to the end, as the participants in this study did. So-called normal computer users tend to scan through a manual in attempts to find instructions and examples that explain the procedures they need or think they need. They are likely to ignore steps and sections of an instructional manual or a training program that seem irrelevant to their task-oriented concerns (Carroll et al., 1988). Thus, the mode of study the participants were asked to adapt in this study can be considered unusual. A follow-up study that allows participants to use the manual in any way they choose will provide another test of the relative effects of active learning and type of elaboration on procedure acquisition.

The apparent trade-off between training time and test time clearly needs to be examined under a variety of training conditions, testing conditions, and using a variety of types of learners (e.g., different backgrounds, working memory capacities). If a longer training time leads to even larger and reliable performance differences over an extended period, then the cost of longer training would seem to be a small one.

Appendix A. Sample database with subset of data from various tables used in test phase

FACULTY Table

ID	NAME	SEX	DEPARTMENT
030	TOM KARR	M	MODERN LANGUAGE
034	DAVID LANDERS	M	PE

TEACHING Table

ID	COURSE	TITLE	HOUR
051	MATH203	CALCULUS III	5
030	FRENCH203	INTERMEDIATE FRENCH III	4
077	HIST101	INTRO TO AMERICAN HISTORY I	3

STUDENT Table

ID	SEX	NAME	HOMESTATE	MAJOR	YEAR	STANDING
367	M	ERIC ANDERSON	CA	ENGINEERING	4	HONOR
370	M	CARL MCKATHY	SC	COMPUTER	1	GOOD
371	F	MARY GORDON	GA	HISTORY	2	PROBATION

TAKING Table

ID	COURSE	SECTION	HOUR
349	MATH303	A	3
349	FRENCH203	B	4
349	EG201	A	2
349	PHYSICS203	A	5

Appendix B. Sample page from SQL manual for the Condition Elaboration Active (C-ACT) Group

Lesson 4: Selecting Specific Columns

If you do not want to display all columns contained in a table, list only the names of those columns you want to display in your SELECT clause.

EXAMPLE:

```
SELECT    DEPTNO, DNAME
FROM      DEPT;
```

Computer Will Display:

DEPTNO	DNAME
10	ACCOUNTING
20	RESEARCH
30	SALES
40	OPERATIONS

EXERCISE:

Fill in the blanks to complete the following statements about the example query above.

You will issue this query if you wish to list the columns _____ and _____ but not the column _____ from the _____ table.

Answer (appeared on the next page of the manual)

You will issue this query if you wish to list the columns DEPTNO and DNAME but not the column LOC from the DEPT table.

Note. Participants were able to refer to the example database (see Appendix F) in order to study the examples or answer the structured exercises; thus they could determine which column was NOT to be retrieved from the DEPT table with the above query.

Appendix C. Sample page from SQL manual for the Condition Elaboration Passive (C-PAS) Group

Lesson 4: Selecting Specific Columns

If you do not want to display all columns contained in a table, list only the names of those columns you want to display in your SELECT clause.

EXAMPLE:

You will issue this query if you wish to list the columns DEPTNO and DNAME but not the column LOC from the DEPT table.

```
SELECT    DEPTNO, DNAME
FROM      DEPT;
```

Computer Will Display:

DEPTNO	DNAME
10	ACCOUNTING
20	RESEARCH
30	SALES
40	OPERATIONS

Appendix D. Sample page from SQL manual for the Condition–Action Pairing Active (CA-ACT) Group

Lesson 4: Selecting Specific Columns

If you do not want to display all columns contained in a table, list only the names of those columns you want to display in your SELECT clause.

EXAMPLE:

```
SELECT    DEPTNO, DNAME
FROM      DEPT;
```

Computer Will Display:

DEPTNO	DNAME
10	ACCOUNTING
20	RESEARCH
30	SALES
40	OPERATIONS

EXERCISE:

Fill in the blanks to complete the following statements about the example query above.

If you wish to display two columns, department number and department name, but not location, the SELECT clause should be: SELECT _____, _____. Since these data values are found in the DEPT table, the FROM clause should be: FROM _____.

Answer (appeared on the next page of the manual)

If you wish to display two columns, department number and department name, but not location, the SELECT clause should be: SELECT DEPTNO, DNAME. Since these data values are found in the DEPT table, the FROM clause should be: FROM DEPT;.

Appendix E. Sample page from SQL manual for the Condition–Action Pairing Passive (CA-PAS) Group

Lesson 4: Selecting Specific Columns

If you do not want to display all columns contained in a table, list only the names of those columns you want to display in your SELECT clause.

EXAMPLE:

If you wish to display two columns, department number and department name, but not location, the SELECT clause should be: SELECT DEPTNO, DNAME. Since these data values are found in the DEPT table, the FROM clause should be: FROM DEPT;.

```
SELECT    DEPTNO, DNAME
FROM      DEPT;
```

Computer Will Display:

DEPTNO	DNAME
10	ACCOUNTING
20	RESEARCH
30	SALES
40	OPERATIONS

Appendix F. Database and description used in training examples and exercises

The following is an example database consisting of two tables. DEPT is a sample table containing information about the departments of a company, and EMP is a sample table containing information about employees who work for that company.

DEPT table

Column Names	Descriptions
DEPTNO:	Department identification number
DNAME:	Department name
LOC:	Department location

DEPTNO	DNAME	LOC
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON

EMP table

Column Names	Descriptions
EMPNO:	Employee identification number
ENAME:	Employee's name
JOB:	Employee's job/title
SAL:	Employee's salary
DEPTNO:	Department identification number

EMPNO	ENAME	JOB	SAL	DEPTNO
7369	SMITH	CLERK	800.00	20
7499	ALLEN	SALESMAN	1600.00	30
7521	WARD	SALESMAN	1250.00	30
7566	JONES	MANAGER	2975.00	20
7654	MARTIN	SALESMAN	1250.00	30
7698	BLAKE	MANAGER	2850.00	30
7782	CLARK	MANAGER	2450.00	10
7788	SCOTT	ANALYST	3000.00	20
7839	KING	PRESIDENT	5000.00	10
7844	TURNER	SALESMAN	1500.00	30
7876	ADAMS	CLERK	1100.00	20
7900	JAMES	CLERK	950.00	30
7902	FORD	ANALYST	3000.00	20
7934	MILLER	CLERK	1300.00	10

Appendix G. Hints for different types of errors

Colon After SELECT/FROM/WHERE (a type of trivial error)

H1: You used an unnecessary punctuation in the SELECT/FROM/WHERE clause.
H2: You used an unnecessary punctuation right after the SELECT/FROM/WHERE command.
H3: You should not use a colon after the SELECT/FROM/WHERE command.

Incorrect Command: SEARCH Instead of SELECT (a type of syntax error)

H1: You used an incorrect command to start the query.
H2: The command, SEARCH, is incorrect in SQL.
H3: You should use SELECT to start the query.

Missing a WHERE Clause (an error resulting from imperfect knowledge)

H1: You missed one clause the task description required.
H2: You need a WHERE clause.
H3: You need to start a WHERE clause by typing "WHERE" and then specify the restricting condition.

Listing All Column Names Instead of Using an Asterisk as Shorthand (a non-optimal action)

H1: You should use a shorthand to list all columns in a table.
H2: You should use an asterisk to list all the columns in the table XXX.

References

Anderson, J. R. (1983). *The architecture of cognition*. Cambridge, MA: Harvard University Press.

Atkinson, R. K., Catrambone, R., & Merrill, M. M. (2003). Aiding transfer in statistics: Examining the use of conceptual equations and elaborations during subgoal learning. *Journal of Educational Psychology*, *95*, 762–773.

Atkinson, R. K., Derry, S. J., Renkl, A., & Wortham, D. W. (2000). Learning from examples: Instructional principles from the worked examples research. *Review of Educational Research*, *70*, 181–214.

Bovair, S., & Kieras, D. (1991). Toward a model of acquiring procedures from text. In R. Barr, M. L. Kamil, P. Mosenthal, & P. D. Pearson (Eds.), *Handbook of reading research* (pp. 206–229). New York: Longman.

Carroll, J. M. (Ed.). (1998). *Minimalism beyond the Nurnberg Funnel*. Cambridge, MA: MIT Press.

Carroll, J. M., & Kay, D. S. (1985). Prompting, feedback and error correction in the design of a scenario machine. In *Proceedings of CHI '85* (pp. 14–30). New York: ACM.

Carroll, J. M., Smith-Kerker, P. L., Ford, J. R., & Mazur-Rimetz, S. A. (1988). The minimal manual. *Human–Computer Interaction*, *3*, 123–153.

Catrambone, R. (1996). Generalizing solution procedures learned from examples. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, *22*, 1020–1031.

Catrambone, R. (1998). The subgoal learning model: Creating better examples so that students can solve novel problems. *Journal of Experimental Psychology: General*, *127*, 355–376.

Catrambone, R., & Carroll, J. M. (1987). Learning a word processing system with training wheels and guided exploration. In *Proceedings of CHI+GI human factors in Computing Systems and Graphics Interface Conference* (pp. 169–174). New York: ACM.

Chi, M. T. H., Bassok, M., Lewis, M. W., Reimann, P., & Glaser, R. (1989). Self-explanations: How students study and use examples in learning to solve problems. *Cognitive Science*, *13*, 145–182.

Chi, M. T. H., & VanLehn, K. A. (1991). The content of physics self-explanations. *The Journal of the Learning Sciences*, *1*, 69–105.

Dufresne, R. J., Gerace, W. J., Hardiman, P. T., & Mestre, J. P. (1992). Constraining novices to perform expertlike problem analyses: Effects on schema acquisition. *The Journal of the Learning Sciences*, *2*, 307–331.

Gerjets, P., Scheiter, K., & Catrambone, R. (2004). Designing instructional examples to reduce intrinsic cognitive load: Molar versus modular presentation of solution procedures. *Instructional Science*, *32*, 33–58.

Introduction to SQL. (1987). Belmont, CA: ORACLE.

Jacoby, L. L. (1978). On interpreting the effects of repetition: Solving a problem versus remembering a solution. *Journal of Verbal Learning and Verbal Behavior*, *22*, 649–667.

Kalyuga, S., Chandler, P., & Sweller, J. (1998). Levels of expertise and instructional design. *Human Factors*, *40*, 1–17.

Mayer, R. E. (2001). *Multimedia learning*. New York: Cambridge University Press.

Merrill, D. C., Reiser, B. J., Merrill, S. K., & Landes, S. (1995). Tutoring: Guided learning by doing. *Cognition and Instruction,* 315–372.

Merrill, D. C., Reiser, B. J., Ranney, M., & Trafton, J. G. (1992). Effective tutoring techniques: A comparison of human tutors and intelligent tutoring systems. *The Journal of the Learning Sciences*, *2*, 277–306.

Mitrovic, A. (2003). An intelligent SQL tutor on the web. *International Journal of Artificial Intelligence in Education*, *13*, 171–195.

Pirolli, P., & Bielaczyc, K. (1989). Empirical analyses of self-explanation and transfer in learning to program. In *Proceedings of the 11th annual conference of the Cognitive Science Society* (pp. 450–457). Hillsdale, NJ: Lawrence Erlbaum Associates.

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

Renkl, A. (1999). Learning mathematics from worked-out examples: Analyzing and fostering self-explanations. *European Journal of Psychology of Education*, *14*, 477–488.

Smelcer, J. B. (1989). *Understanding user errors in database query*, Unpublished doctoral dissertation, University of Michigan, Ann Arbor, MI.

Sweller, J. (1988). Cognitive load during problem solving: Effects on learning. *Cognitive Science*, *12*, 257–285.

Sweller, J., van Merriënboer, J. J. G., & Paas, F. (1998). Cognitive architecture and instructional design. *Educational Psychology Review*, *10*, 251–296.

Welty, C. (1985). Correcting user errors in SQL. *International Journal of Man–Machine Studies*, *22*, 463–477.