Testing a Walkthrough Methodology for Theory-Based Design of Walk-Up-and-Use Interfaces

Clayton Lewis, Peter Polson, Cathleen Wharton, and John Rieman

Institute of Cognitive Science
Campus Box 430
University of Colorado, Boulder CO 80309
303-492 6657, clayton@sigi.colorado.edu

ABSTRACT
The value of theoretical analyses in user interface design has been hotly debated. All sides agree that it is difficult to apply current theoretical models within the constraints of real-world development projects. We attack this problem in the context of bringing the theoretical ideas within a model of exploratory learning [19] to bear on the evaluation of alternative interfaces for walk-up-and-use systems. We derived a "cognitive walkthrough" procedure for systematically evaluating features of an interface in the context of the theory. Four people independently applied this procedure to four alternative interfaces for which we have empirical usability data. Consideration of the walkthrough sheds light on the consistency with which such a procedure can be applied as well as the accuracy of the results.

KEYWORDS: Design methodology, formal models of human computer interaction, walk-up-and-use systems.

INTRODUCTION

Application of formal cognitive models of human-computer interaction to the design of computing systems within the constraints imposed by an actual development process is a hotly debated question [3,4,6]. Some have claimed that the cognitive processes described in our current models of human-computer interaction do not provide guidance for a wide enough range of issues to be of any real use in the development process [6,7,23]. Also, there is widespread agreement that applying cognitive models to evaluate a design is a difficult task [2,3,10]. Applying formal cognitive theory usually involves developing a simulation model of the processes involved in performing a task using the proposed design [16]. Often, building the model is equivalent in complexity to programming the proposed application.

The issue we explore in this paper is how one extracts design and evaluation guidance from a formal theory of human-computer interaction. We accept the argument that building complete simulation models to evaluate a design is not feasible in a typical development process. In this paper, we describe how theory may be used to guide a design review. The methodology we propose is analogous to the kinds of structured walkthroughs advocated by various individuals in the software engineering community [8,22,24].

We use a theory of exploratory learning developed by Polson and Lewis [19] to generate a list of theoretically motivated questions about the user interfaces of systems intended for use in applications with minimal formal training requirements, e.g. advanced phone services and other consumer applications of computer technology. The questions focus on the interaction between the design and a user attempting to perform a specific task. Positive responses to individual questions support the inference that the interface will be easily learned. Negative responses highlight steps in an operating procedure that will be difficult to learn and suggest the causes for a potential source of difficulty.

The remainder of the paper is organized as follows. In the second section, we present a brief summary of the Polson and Lewis [19] model of exploratory learning, CE+. We then discuss the design guidelines that Lewis and Polson derived from CE+ and outline problems with these guidelines. Our arguments are very similar to general kinds of criticism that have been leveled against all kinds of design guidelines, for example those in Smith and Mosier [21] and Rubenstein and Hersh [20]. In the fourth section, we present the details of our cognitive walkthrough procedure, discuss potential advantages, and compare the walkthrough process with design guidance derived from theoretically motivated guidelines. In the next section we present an evaluation of the methodology, in which three out of the four authors of this paper did independent walkthroughs of four different user interface designs for which we have extensive user test data. The final section offers general conclusions as to the value of the methodology.

THE CE+ MODEL OF EXPLORATORY LEARNING

Polson and Lewis [19] develop a cognitive theory of initial learning in human-computer interaction and derive from the theory a set of design guidelines to support the development of applications requiring minimal learning on the part of users. The resulting model is similar to ACT* [1] and SOAR [12]. The model contains a problem-solving component, a learning component, and an execution component. It combines a rule-based representation of procedural knowledge from Cognitive Complexity Theory [11,16], the analysis of system responses from Lewis's work on learning from demonstrations [13], and a model of the problem-solving processes derived from the puzzle-problem literature of the late '70s and early '80s [18].

The problem-solving component of CE+ predicts that a user will choose among alternative actions based on the similarity between the user's expectation of the consequences of an action and the user's current goal. The comparison process is strongly influenced by the superficial similarity of the descriptions of goal and actions. A match between and command name and a term or phrase in the goal description can cause a user to select that command. The problem-solving process is analogous to hill-climbing, which is a variety of means-ends analysis. This type of problem-solving behavior has been found in a large number of studies on subjects working in novel problem domains [9].

After the selected action has been executed, the user evaluates the response provided by the system and makes a decision as to whether or not progress is being made toward the goal. Progress evaluation is done using heuristics proposed by Lewis [13] to model learning from demonstrations. It is of particular importance that a user receives system responses that he perceives as being related to the original goal. If a mismatch is detected, the user will attempt to undo the just taken action.

The learning occurs when the evaluation process leads to a positive decision. The previously taken step is stored in memory in the form of a rule. Thus, Polson and Lewis [19] follow Anderson in concluding that the major problems in acquiring skills are due to the difficulty and complexity of the problem-solving processes and not to the encoding processes that store successful problem-solving episodes in long term memory.

The execution component of CE+ models the user by first attempting to fire an applicable rule that matches the current context. If none is found, the problem-solving component described above is invoked and the model attempts to discover an action that leads to a positive evaluation of progress.

DESIGN-FOR-SUCCESSFUL-GUESSING

Polson and Lewis [19] derive, from the model that we have briefly outlined above, a set of design principles. They argue that the knowledge-poor problem-solving strategies used by the theory are a guessing process; thus they called their principles 'Design-for-Successful-Guessing.' As an example, four out of the eight design principles from their paper are as follows:

1. Make the repertory of available actions salient.
2. Provide an obvious way to undo actions.
3. Offer few alternatives.
4. Require as few choices as possible.

In the following,we briefly describe the rationale for two of the guidelines. Guideline 1, "Make the repertory of available actions salient," makes the obvious point that the theory assumes that new users explicitly evaluate possible actions by comparing each to the goal. If an action is unknown to the user, obviously it will never be considered or executed. A good illustration of this error occurred in an early version of the IBM Displaywriter, a menu based system, that used a special key to summon a critical auxiliary menu in order to invoke the print command. When new users were given the task of printing a document, they diligently searched through the menu hierarchy. However, they had no information that would enable them to deduce that they had to press a critical key which would make the print option available to them.

Guideline 2, "Provide obvious ways to undo actions," is necessary because of the nature of the learning process. Users are making a series of guesses about what the appropriate next action should be. This trial and error problem-solving process may lead to a fair number of mistakes. The interface has to provide both the feedback necessary for the user to detect such mistakes and a safe and obvious way to undo erroneous actions.

We briefly discuss two difficulties with such guidelines: applicability and tradeoffs. Guideline 3, "Offer few alternatives," suggests constraining each choice to a few alternatives. Guideline 4, "limit the number of choices," makes both the problems of applicability and tradeoffs very clear. These guidelines suggest that an interface should have both a narrow and shallow menu structure. If the interface must support a large number of alternative commands, this is obviously impossible. A more complete analysis involves examining the process by which a user attempts to discriminate between a large number of alternatives. If it can be assumed that users understand differences between a large number of tasks — 10 or 15 — and the menu items clearly described these differences, the theory predicts that users would be able to select the correct alternative out of 10 to 15 choices. Thus, a guideline's applicability or the tradeoffs between conflicting guidelines are resolved by evaluating the design using the theory that motivated the guidelines. In the next section of the paper we describe a methodology for such an evaluation.

CE+ Design Walkthrough **Evaluator** _____ **Date** ____

Interface _____ **Task** _____ **Step #** ___

Actions/choices should be ranked according to what percentage of potential users are expected to have problems: 0 = none; 1 = some; 2 = more than half ; 3 = most.

1. Description of user's immediate *goal:*
2. (First/next) atomic *action* user should take:
    2a. Obvious that action is *available* ? Why/Why not?
    2b. Obvious that action is *appropriate to goal* ? Why/Why not?
3. How will user *access description* of action?
    3a. Problem accessing? Why/Why not?
4. How will user *associate* description with action?
    4a. Problem associating? Why/why not?
5. All other available actions *less appropriate* ? For each, why/why not?
6. How will user *execute* the action?
    6a. Problems? Why/why not?
7. If *timeouts,* time for user to decide before timeout? Why/why not?
8. Execute the action. Describe *system response:*
    8a. Obvious *progress* has been made toward goal? Why/why not?
    8b. User can access needed *info. in system response?* Why/why not?
9. Describe appropriate *modified goal,* if any:
    9a. Obvious that *goal should change?* Why/why not?
    9b. If task *completed,* is it obvious? Why/why not?

Figure 1. A cognitive walkthrough evaluation form for a single action

THE COGNITIVE WALKTHROUGH

The cognitive walkthrough is a theoretically structured evaluation process that takes the form of a list of questions (Figure 1). The questions focus the designer's attention on individual aspects of the interface that the CE+ theory claims are important in facilitating the problem-solving and learning processes. The overall process goes as follows.

First, the designer (or design team) specifies a series of tasks on which one will evaluate the design. Next, the sequence of user actions that will successfully perform a given task is specified by the designer. There may be more than one such sequence. Finally, the main part of the walkthrough involves using the problem-solving and feedback evaluation processes from CE+ to evaluate the ease of learning for the proposed design on a particular task. If an evaluator expects no problems at a given step, that judgment has to be defended. If problems are expected, they should be described.

The evaluation of each step in a task includes the following processes. The evaluator begins by giving a description of the user's current goals and the correct action (Figure 1, Questions 1 and 2). The next series of questions (Questions 2a through 7) evaluate the ease with which the user will be able to correctly select that action and execute it. Next, the evaluator describes the system response and judges its adequacy (Question 8). The final question (Question 9) evaluates whether the user's ability to form an appropriate goal for the next action or detect that the task has been completed. If the task is not complete, the evaluator assumes that the goals have been correctly modified and proceeds to evaluate the next step.

A cognitive walkthrough makes explicit the important design decisions that have been made either implicitly or explicitly in the process of designing an interface and the implications of those decisions for the problem-solving processes and learning by exploration. In essence, the designer is doing a hand simulation of the processes described by the theory that are involved in action selection. The processes by which a user attempts to learn a system involve complex interactions between the cognitive processes of the user, the characteristics of the tasks, and the details of a particular user interface. Making correct inferences about the usability of an interface requires an understanding of the roles of mental processes, especially action selection and goal transformations.

A detailed understanding of these interactive processes is, of course, one of the major advantages of doing a complete simulation of the processes involved in using an interface to perform a specific task. The disadvantages, mentioned previously, are obvious. Building a simulation model is a task equivalent in difficulty to developing a small or moderate sized expert system. Our goal in developing the cognitive walkthrough methodology described above is to achieve much of the detailed understanding of interactive processes provided by a simulation model without investing the resources necessary to develop a complete running simulation.

THE EFFECTIVENESS OF COGNITIVE WALKTHROUGHS

In this section we describe the results achieved by evaluating an interface using the cognitive walkthrough methodology. We compare those results to empirical data for the same interface. Two major issues were whether the technique would give consistent results across evaluators and whether the walkthrough would predict the same problems identified by the empirical data.

Description of the Interface and Tasks Evaluated

Both the walkthrough and the empirical study evaluated four different interface designs to a mail messaging system. The system allowed users to read messages, more forward and backward among messages, and delete messages. There were two major differences between the interfaces. First, in the Model versions (Model No-Feedback [MNF] and Model Feedback [MF]), playing or deleting a message advanced the system pointer to the next message, whereas the Explicit versions (Explicit No-Feedback [ENF] and Explicit Feedback [EF]) required the user to explicitly press the FORWARD key to advance the message pointer. Second, the No-Feedback versions (MNF and ENF) supplied no information as to where in the message list the pointer was currently located, whereas the Feedback versions (MF and EF) did provide this information. Feedback versions also informed the user when a message had been played or deleted.

In all versions, the commands available were PLAY, DELETE, FORWARD, BACKWARD, DONE, and RESET. The DONE command was used to terminate a session after the user had finished a task successfully. The RESET command allowed a user to restart a task. Pressing DONE when the task had not been successfully completed had the same effect as pressing RESET.

Two tasks were evaluated. Task 1 was to "Play all messages," and three messages were provided. Task 2 was to "Play all messages and then delete them," and four messages were provided. The Model versions of the interface, especially Model No-Feedback, were designed to be especially difficult for the second task, since pressing PLAY (or DELETE) had the dual effect of playing (or deleting) a message and moving the message pointer. Where the task was to play and then delete several messages, pressing PLAY followed by DELETE in either of the Model versions would play the first message and delete the second, a situation from which the user could only recover by resetting the interface and starting over.

Description of the Walkthrough

Four evaluators each performed independent cognitive walkthroughs of Tasks 1 and 2, for all four interfaces to the mail messaging system. Three of the four evaluators (C.L., C.W., J.R.) were familiar with the CE+ model. These evaluators also had discussed the general trends of

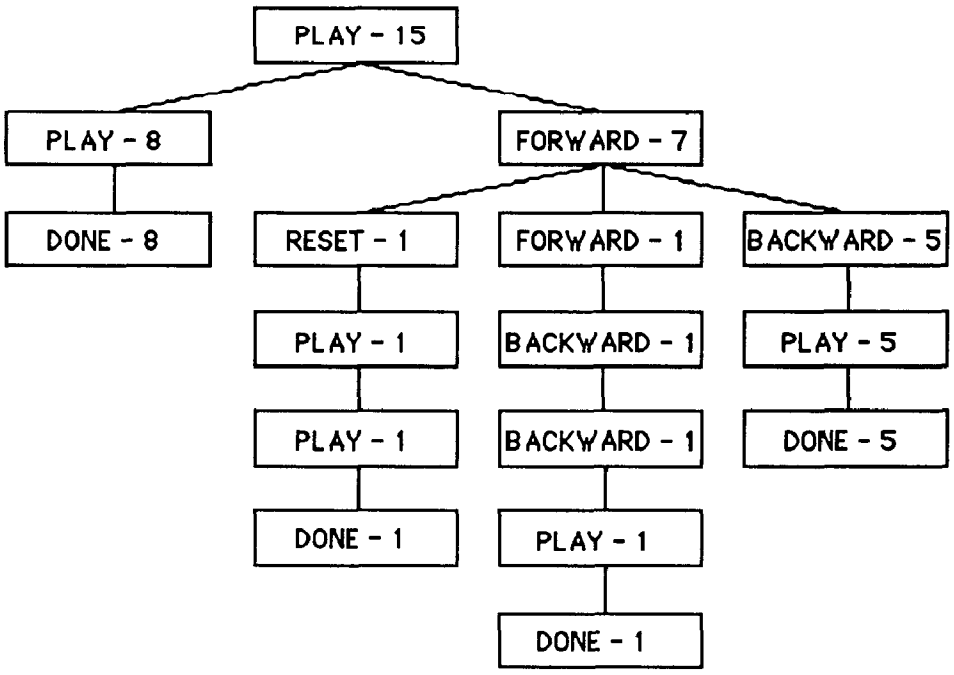

Figure 2. Example of an action tree for the task "Play all your messages," with two messages.

data already gathered by an empirical study of the same interfaces, but none had evaluated the data at the level of detail that the walkthrough would predict. The fourth evaluator was not familiar with CE+ or the data. Each evaluator was instructed to explore each interface, determine a solution path (i.e., action sequence), and then perform the walkthrough. Problems identified during the walkthrough were to be rated according to a discrete scale: 0 = no users would have problems; 1 = some would have problems; 2 = more than half would have problems; 3 = most users would select the wrong action.

In addition to the main walkthrough procedure, three of the evaluators considered the robustness of each interface as predicted by the theory. The question here was whether the interface would help or hinder a user who had deviated from the direct path to a solution.

Consistency Across Evaluators

After all the walkthroughs had been completed, two of the evaluators (C.W. and J.R.) compared the results. It was found that 20 unique problems had been identified across all evaluations, of which 13 had been noted by three or more of the evaluators, and 18 had been noted by two or more of the evaluators. The three evaluators who were familiar with the CE+ theory had agreed on 9 of the 20 problems, whereas the fourth evaluator had noted only one of these problems. Where more than one evaluator noted the same problem, there was often disagreement as to the numerical rating and as to where on the evaluation form the problem should be recorded: e.g., as a incorrectly formed goal (Question 1 and Question 9) or as an inappropriate response to a correct but unclear goal (Question 5). For the robustness analysis, the three evaluators familiar with the theory had ranked the interfaces in the same order.

Walkthrough Results Compared to Empirical Results

As part of a larger project, empirical data had been collected for the two tasks with each of the four interfaces. Each interface-task combination had been tested on at least 15 subjects. These results had been compiled by the experimenters into action trees. Action trees are graphic representations of the paths taken by subjects or groups of subjects through the space of actions afforded by the interface. An example of an action tree for a simple task is shown in Figure 2. In the experiment recorded in this tree, 15 users pressed PLAY as their first action, then eight pressed PLAY once more, followed by DONE. Seven users pressed FORWARD after the first PLAY, then deviated into various other paths.

Using the action trees, we compared the empirical data to a list of 18 problems agreed on by two or more of the evaluators. We looked for points in the trees where subjects had deviated from a direct path to a solution, and where that deviation seemed to reflect one of the problems identified in the walkthroughs. Of the 18 problems, evidence for 15 was found in the data. The remaining three were of a type that the data could not reflect (e.g., one evaluator predicted that a user might expect audio feedback, but no specific error response was predicted).

Table 1. Empirical results compared with walkthrough predictions

Condition/ Task[1]	Possible Paths[2]	Paths Predicted by Walkthrough[3]	Observed Paths[4] (predicted/total)	Observed Subject Traversals[5] (predicted/total)	Observed Paths to Error[6] (predicted/total)	Observed Subject Traversals to Error[7] (predicted/total)
MNF / 1	25	5	2 / 3	14 / 15	1 / 2	6 / 7
MF / 1	25	5	1 / 5	7 / 15	0 / 4	0 / 8
ENF / 1	37	9	1 / 4	12 / 16	1 / 4	12 / 16
EF / 1	37	9	4 / 6	12 / 15	3 / 5	11 / 14
MNF / 2	937	17	2 / 5	8 / 15	2 / 5	8 / 15
MF / 2	937	6	0 / 8	0 / 17	0 / 8	0 / 17
ENF / 2	777	11	5 / 9	10 / 16	4 / 8	8 / 14
EF / 2	777	6	3 / 11	7 / 15	2 / 8	6 / 14
Totals:	3552	68	18 / 51	70 / 124	13 / 44	51 / 105

[1]MNF = Model No-Feedback; MF = Model Feedback; ENF = Explicit No-Feedback; EF = Explicit Feedback.
[2]Counts of possible paths through the interface for each task, terminating a path at successful completion or the first error.
[3]Paths predicted by evaluators. Includes all paths indicated correct by any evaluator and error paths predicted by two or more evaluators.
[4]Number of distinct paths observed in subject data, terminating a path at successful completion or the first error.
[5]Number of traversals of paths by subjects. Thus, one path traversed by three subjects contributes three traversals.
[6]Number of distinct paths ending in an error observed in subject data.
[7]Number of subject traversals of paths ending in error; see note 5 above for definition of 'subject traversals.'

The two evaluators who evaluated the walkthroughs for consistency also used the walkthrough results to create trees of predicted subject action paths. The walkthrough trees contained (1) any path that one or more evaluator had identified as a direct path to a solution, and (2) any path that proceeded along an identified solution path and then branched into an error action predicted by two or more evaluators. Possible error or recovery paths beyond any first branch off a direct path were not included in the trees.

The trees predicted by the walkthroughs were evaluated according to two criteria. Obviously, we were interested in locating error actions in the empirical data that matched error actions predicted by the walkthroughs. However, we also wanted to demonstrate that the predictions had not merely encompassed so much of the potential action space that matching error actions was inevitable. As a measure of this problem, we compared the number of paths in the walkthrough trees to the number of possible paths in the action space, where possible paths were defined in the same manner as the predicted action paths in the walkthrough trees. (Note that we are considering only a defined subset of possible action sequences that subjects could generate, since we terminate all paths as soon as an error action is performed.)

The results of matching the walkthrough trees to the empirical action trees are detailed in Table 1. As indicated by the first two columns of the table, the evaluators had confined their predictions to a relatively small subset of all paths available to the subjects, never more than 24 percent (i.e., 9/37) and as low as 0.6 percent (6/937). Nonetheless, the predicted paths included 35 percent (18/51) of the distinct paths traversed by subjects. If we consider the number of subjects traversing the paths, so that more frequent paths are counted more heavily, coverage is better: if we call a traversal of a given path by a subject a subject traversal, 70 out of 124 subject traversals were predicted, or 56.5 percent.

Focusing on errors, of 44 paths leading to errors that were observed, 13 were predicted. Taking into account the number of subjects traversing these paths, 51 of 105 subject traversals leading to errors were predicted, or 48.5 percent.

SUMMARY

The results we have presented in this paper show that our cognitive walkthrough methodology can detect almost 50 percent of the problems that were revealed by a full scale evaluation study using two different realistic tasks on a simplified messaging system. The reader should understand that we are using a very strict criterion for the notion of error. Any deviation from a solution path identified in the walkthrough was tallied as a problem. Some of these deviations seem to be due to the importation of background knowledge from phone messaging systems: for example, beginning the session by pressing the BACK key, perhaps to 'rewind' the messages before playing them. Others may

have been due to the simple desire on the part of subjects to explore the user interface before completing the task.

We do not want to claim that the cognitive walkthrough methodology will eliminate the need for evaluating prototypes of the interface. Our arguments are that the walkthrough with a very limited investment in resources, approximately an hour per task per interface, can detect almost 50 percent of the problems encountered by users of the design.

There were inconsistencies between our evaluators that turned out to be quite revealing. One of the four evaluators had read the Polson and Lewis [19] paper but was not deeply familiar with the theory. The other three evaluators had been deeply involved in the development of the theory and simulation models derived from the theory. There was a high level of agreement among these three evaluators, but less with the fourth evaluator. The fourth evaluator predicted fewer observed error paths than the other evaluators, besides making some different predictions. Our claim is that the evaluation process is akin to a hand simulation of the mechanisms described by the theory of exploratory learning. Thus, it would seem reasonable that successful execution of the walkthrough methodology would require deep knowledge of the theory. This result, however, does suggest that successful transfer of the walkthrough methodology to another group would require development of a suitable training program in the underlying theoretical models, including practice exercises to shape intuitions necessary to successfully apply the theory in the walkthrough.

We think the results we have obtained support our basic claims for the properties and effectiveness of cognitive walkthroughs, in the limited context of simple interfaces and tasks, and limited background knowledge by users. Caution in extending the method is suggested by the fact that the proportion of error traversals predicted declines from .64 for Task 1 to .37 for the more complex Task 2. On the other hand, we see no reason in principle why the method cannot be applied in more complex situations, or for that matter adapted to reflect a more complex underlying theory. For example, given a theory which accounted for memory load effects, which CE+ does not, the walkthrough procedure could be expanded to call for assessment of memory load at each stage of an interaction.

ACKNOWLEDGEMENTS

We wish to thank Roland Hübscher for serving as one of our evaluators in the walkthrough. Catherine Ashworth and Douglas Lhotka collected the data that we used to evaluate the predictions from the walk-through. This research was supported by the sponsored research program of US West Advanced Technology and the National Science Foundation under NSF grant IRI 8722792.

REFERENCES

1. Anderson, J.R. Skill acquisition: Compilation of weak-method solutions. *Psychological Review*, 94 (1987), pp. 192-211.

2. Bennett, J., Lorch, D., Kieras, D. E., and Polson, P. G. Developing a user interface technology for use in industry. *Proceeding of Interact87, 2nd IFIP Conference on Human-Computer Interaction* (Stuttgart, Sept. 1987).

3. Butler, K., Bennett, J., Polson, P.G., and Karat, J. Report of the Workshop on Analytical Models: Predicting the Complexity of Human-Computer Interaction. *SIGCHI Bulletin*, 20 (1989), pp. 63-79.

4. Card, S.K. and Newell, A. The prospects for psychological science in human-computer interaction. *Human-Computer Interaction*, 1 (1985), pp. 209-242.

5. Card, S.K., Moran, T.P., and Newell, A. *The Ppsychology of Human-Computer Interaction.* Erlbaum, Hillsdale, N.J., 1983.

6. Carroll, J.M. and Campbell, R.L. Softening up hard science: Reply to Newell and Card. *Human-Computer Interaction*, 2 (1986) pp. 227-250.

7. Carroll, J.M. and Kellogg, W.A. Artifact as Theory-Nexus: Hermeneutics Meets Theory-Based Design. K.Bice and C. Lewis (Eds) *Proceedings of the Conference on Human Factors in Computing Systems, CHI'89.* American Association for Computing Machinery, New York (1989), pp. 7-14.

8. Fagan, M.E. Advances in Software Inspections. *IEEE Transactions on Software Engineering*, SE-12 (1986), pp. 744-751.

9. Greeno, J.G. and Simon, H.A. Problem Solving and Reasoning. In R.C. Atkinson, R. Herrnstein, G. Lindzey, and R.D. Luce (Eds.), *Steven's Handbook of Experimental Ppsychology.* John Wiley and Sons, New York, 1988.

10. Kieras, D.E. Towards a Practical GOMS Model Methodology for User Interface Design. In M. Helander (Ed.) *The Handbook of Human-Computer Interaction.* North-Holland, Amsterdam, NV, 1988.

11. Kieras, D.E. and Polson, P.G. An approach to the formal analysis of user complexity. *International Journal of Man-Machine Studies*, 22 (1985), pp. 365-394.

12. Laird, J., Newell, A., and Rosenbloom, P. SOAR: An architecture for general Intelligence. *Artificial Intelligence*, 33 (1987), pp. 1-64.

13. Lewis, C.H. How and why to learn why: Analysis-based generalization of procedures. *Cognitive Science*, 12 (1988), pp. 211-256.

14. Norman, D.A. Cognitive Engineering. In D.A. Norman and S.W. Draper (Eds.) *User Centered Systems Design: New Perspectives in Human-Computer Interaction.* Lawrence Erlbaum Assoc., Hillsdale, NJ, 1986

15. Norman, D.A. *The Psychology of Everyday Things.*: Basic Books, New York, NY, 1988.

16. Polson, P.G. A quantitative theory of human-computer interaction. In J.M. Carroll (Ed.). *Interfacing thought: Cognitive Aspects of Human-Computer Interaction.* Bradford Books/MIT Press, Cambridge, MA, 1987.

17. Polson, P.G. Transfer and Retention. In R. Guindon (Ed.), *Cognitive Science and Its Application for Human-Computer Interaction.* Lawrence Erlbaum Assoc., Hillsdale, NJ, 1988.

18. Polson, P. G. and Jeffries, R. Problem solving as search and understanding. In R. J. Sternberg (Ed.) *Advances in the Psychology of Human Intelligence*, Vol. I. Lawrence Erlbaum Associates, Hillsdale, N.J., 1982, pp. 367-412.

19. Polson, P.G. and Lewis, C.H. Theory-Based Design for Easily Learned Interfaces. *Human Computer Interaction.* (In press).

20. Rubenstein, R. and Hersh, H.M. *The Human factor: Designing Computer Systems for People.* Digital Press, Burlington, MA, 1984.

21. Smith, S.L. and Mosier, J.N. Guidelines for designing the user interface software. Report 7 MTR-10090, Esd-Tr-86-278. Mitre Corporation, Bedford, MA, 1986.

22. Weinberg, G.M. and Freedman, D.P. Reviews, Walkthroughs, and Inspections. *IEEE Transactions on Software Engineering*, SE-10, 1984, pp. 68-72.

23. Whiteside, J., Bennett, J., and Holtzblatt, K. Usability Engineering: Our Experience and Evolution. In M. Helander (Ed.) *The Handbook of Human-Computer Interaction.* North-Holland, Amsterdam, NV, 1988, pp. 791-816.

24. Yourdon, E. *Structured Walkthroughs.* (4th ed.). Yourdon Press, Englewood Cliffs, NJ, 1989.