Int. J. Man–Machine Studies (1992) **36**, 741–773

Cognitive walkthroughs: a method for theory-based evaluation of user interfaces

PETER G. POLSON,* CLAYTON LEWIS, JOHN RIEMAN† AND CATHLEEN WHARTON†

*Institute of Cognitive Science, *Department of Psychology and the †Department of Computer Science, University of Colorado, Boulder, CO 80309-0345, USA*

(Received 1 August 1990 and accepted in revised form 8 April 1991)

This paper presents a new methodology for performing theory-based evaluations of user interface designs early in the design cycle. The methodology is an adaptation of the design walkthrough techniques that have been used for many years in the software engineering community. Traditional walkthroughs involve hand simulation of sections of code to ensure that they implement specified functionality. The method we present involves hand simulation of the cognitive activities of a user, to ensure that the user can easily learn to perform tasks that the system is intended to support. The cognitive walkthrough methodology, described in detail, is based on a theory of learning by exploration presented in this paper. There is a summary of preliminary results of effectiveness and comparisons with other design methods.

1. Introduction

It is widely acknowledged that many current computer systems and applications are too hard to master and difficult to use (Whiteside, Bennett, & Holtzblat, 1988; Shackel, 1984). There are many reasons for this, ranging from lack-of-fit of a tool to the social and organizational context (Winograd & Flores, 1986) to shortcomings in the software's user interface, e.g. lack of feedback, complex dialog structures, and hard-to-remember commands (Norman, 1988; Gentner & Grudin, 1990). User interface problems can also be traced to failures in the management of the development process: usability goals are not explicitly specified, and thus their achievement is not managed during development (Bennett, 1984).

1.1. OVERVIEW OF THE METHOD

The design evaluation methodology proposed in this paper, the cognitive walkthrough, aims to provide a new tool for assessing the usability of a system, and assigning causes to usability problems, early in the design process. Our focus in this paper is on ease of learning. There are many situations in which users want to make use of a system without having to invest much time or receive any formal instruction. The method gives designers the ability to anticipate some learnability problems before an implementation or even a mock-up of the design is available, without empirically testing a prototype with representative users.

A cognitive walkthrough applies relevant cognitive theory in an evaluation process that can be used by software developers under the constraints on time and other resources imposed by the typical development process. Bellotti (1990) discusses these constraints. A cognitive walkthrough has the same basic organization and rationale as other types of design walkthroughs, including architecture

0020-7373/92/050741 + 33$03.00/0

walkthroughs, functional requirements walkthroughs, document walkthroughs and code walkthroughs. (Yourdon, 1989; Fagan, 1986). It is a review process in which the author of a particular aspect of a design presents his or her proposed design solution to a group of peers. The peers evaluate the solution using an explicit set of criteria appropriate to the particular classes of design issues. In the cognitive walkthrough, the criteria focus on the cognitive processes needed to perform tasks with the system as designed.

The purpose of an evaluation of a design using the cognitive walkthrough methodology is to evaluate the ease with which users can perform a task with little or no formal instruction or informal coaching. The input to a cognitive walkthrough session includes a detailed design description of the user interface, a task scenario, explicit assumptions about the user population and the context of use, and a sequence of actions with which a user could successfully complete the task using the design under evaluation.

During the walkthrough process, the reviewers step through the actions, considering the behavior of the interface and its effect on the user, and attempting to identify those actions that would be difficult for the average member of the proposed user population to choose or to execute. Claims that a given step would not cause any difficulties must be supported by theoretical arguments, empirical data, or relevant experience and common sense of the team members. Claims that a given step will be problematic must also be supported by relevant theory, data or experience. In essence, performing a walkthrough involves carrying out a hand simulation of the cognitive processes that are required to successfully guess the specified action sequence.

1.2. THE APPLICATION OF COGNITIVE THEORY IN THE DESIGN PROCESS

There have been numerous unsuccessful attempts to apply cognitive theory to design more usable systems (Carroll & Campbell, 1986; Bennett *et al.* 1987; Carroll & Kellogg, 1989). Bellotti (1990) reviews several theoretically-based design methodologies; she found that they are not applied in practice because they are not usable in actual design contexts. One of our major goals in proposing the cognitive walkthrough is to develop a theoretically-based design methodology that can be used in actual development situations. It is based on a methodology that is well understood and actually used by developers, design walkthroughs (Yourdon, 1989).

The cognitive walkthrough is an evaluation methodology that focuses on ease of learning. It is especially appropriate for the development of applications where users must (or prefer) to master a new application or function by learning through exploration. Two examples are walk-up-and-use systems, e.g. ATMs and telephone-based applications, and new functions in a frequently used application. Such systems are particularly difficult and costly to develop successfully. Typical design guidelines, e.g. Smith and Mosier (1986) or Rubenstein and Hersh (1984), provide very general recommendations that are not specific enough to guide the development process. In the past, ease of learning has had to be evaluated empirically using prototypes or early versions of the application. Often, these costly studies were not performed until late in the design cycle, when solutions to serious problems could require extensive revisions and large delays.

1.3. OVERVIEW OF PAPER

We begin by describing the theory of exploration that provides the foundation for our walkthrough methodology. Following this, we describe cognitive walkthroughs and present a detailed example. We then summarize our experience with the method. This is followed by a comparison of the walkthrough methodology with other proposals to apply cognitive theories to design. The last section summarizes our conclusions about the walkthrough methodology and discusses possible extensions.

2. A theory of learning by exploration

A cognitive walkthrough evaluates the ease with which a typical user can successfully perform a task using a given interface design. Our focus in this paper is on tasks the user must learn to do by exploration, that is, by guessing what to do using cues provided by the system, rather than by knowing how to use the system. This evaluation is based on a detailed model of the cognitive processes involved in successful exploration, i.e. correctly guessing the required actions. We extract from this model a set of evaluation criteria that enables the designer to identify points during an interaction sequence where a typical user will succeed or fail, and to characterize causes of failures. The cognitive walkthrough consists of a way of examining a proposed interface systematically in order to identify these failure situations.

2.1 THEORETICAL FOUNDATIONS

Our model of exploration is an extension of a model of learning by exploration proposed by Polson and Lewis (1990). The model is related to Norman's (1986, 1988) theory of action that forms the theoretical foundation for his work on cognitive engineering. Norman's framework specifies a series of stages beginning with the user's initial goal, leading to the generation of a plan for an action, execution of the action, evaluation of the feedback, revision of the goals and the continuation of this cycle. The same basic cyclic model of action generation and evaluation of consequences underlies our model of exploratory learning. Our model also provides some insight into the cognitive mechanisms that control the cycle, allowing specific predictions of certain cases in which failures may occur.

2.2. OUTLINE OF THE MODEL

We have developed our model within the framework of Kintsch's construction–integration model (Kintsch, 1988; Mannes & Kintsch, 1991; Doane, Kintsch & Polson, 1990). The construction-integration model (Kintsch, 1988) describes the processes by which users integrate a representation of text or other perceptual input with background knowledge to construct a representation that will enable them to perform a task. In outline, the model works as follows. An initial goal structure is constructed from a description of the user's task. Goal structures in the model are similar to the goal hierarchies postulated by the GOMS model (Card, Moran & Newell, 1983; Bovair, Kieras & Polson, 1990), with a top goal representing the overall task, intermediate level goals defining a task decomposition and lowest level goals describing individual actions (if these become known).

Goals are represented by propositions. They are linked to other goals, to propositions representing background knowledge, to propositions representing objects seen in the environment and to actions. Activation flows from the top goal along these links to the representations of actions. When an action becomes sufficiently activated, it is executed. Any response by the system is observed and interpreted, resulting in the deactivation of accomplished goals and the building of new propositions. These propositions represent new goals and changes in the environment caused by the last action. These new propositions are linked into the existing network of propositions. Activation now spreads through this new network. The next action occurs when some action becomes sufficiently active, causing the interpretation process to modify the network of propositions, leading to a new cycle.

2.3. LINKS BETWEEN GOALS AND ACTIONS

As just sketched, actions are executed when sufficient activation reaches them. For this to happen there must be a path of associative connections between a user's goal and the representation of the action. A common situation in which such a path exists can be seen in the "label following" strategy employed by naive users (Engelbeck, 1986; Polson & Lewis, 1990). Here an action, such as pressing a button, is chosen because there is a label associated with the action that shares terms with an active user goal. For example, a user with the goal of turning a system off would be expected to press a button labelled "off" if one is available.

The associative path between goal and action is composed of at least four linked propositions: the representations of the user's currently active goal, the button label, the button description including its location, and the action of pressing the button. In the example introduced in the preceding paragraph, the goal proposition is linked to the label proposition because of the shared term 'off' in both goal and label. The label is linked to the button by shared terms that describe their common location. The action of pressing is linked to the particular button by information about the location and knowledge that buttons can be pressed. Activation spreads across these links, and if it reaches a sufficient level, the button will be pressed.

Notice that there are several ways in which this path could be broken, so that the correct action would not be taken: The label may not share terms with the goal. The label might not be placed on the button, which would make the link between label and the button unclear. The user might not know that buttons can be pressed. Even if the path is complete, there are other possible failure modes. For example, there could be two or more labels that seem to be linked to a given goal, and this could cause a competing action to become active before the correct one. Failures of these kinds are checked for in the walkthrough procedure.

Not all actions must be linked to goals in as simple and direct a way as required by the label-following heuristic. However, some added knowledge must be assumed if the user is expected to link an action to a goal without benefit of a label or analogous cue, or expected to link a goal with a label that does not share terms with the goal. The walkthrough procedure helps to make these assumptions explicit. For example, if a button is labelled "1/0" rather than "on/off", a successful user must know what this symbol means. Designers need to consider whether such assumptions are valid for their target user population. Careful attention to questions of

existing knowledge allows the model and the walkthrough procedure to reflect the behavior of expert system users faced with new applications or functionality, as well as novice users who are approaching the system for the first time. (See Doane, Kintsch & Polson, 1990, for further discussion of user knowledge requirements in the context of the construction–integration model.)

2.4. MANAGEMENT OF GOAL STRUCTURE

2.4.1. Generating the goal structure

We assume that the user's initial goals for a task will be incomplete in that they do not specify a complete task decomposition that includes both representations of subtasks and the action sequence necessary to carry out each subtask. Thus, we are not dealing with routine cognitive skills where users have stored in long-term memory a complete and correct representation of the goal structure and the necessary action sequence (Card, Moran & Newell, 1983; Bovair, Kieras & Polson, 1990).

Fragments of the goal structure are generated by the user as he or she interacts with the interface while attempting to carry out the task. The goal generation process will be driven almost exclusively by prompts and other feedback received from the interface. Part of the cognitive walkthrough procedure determines whether the user's background knowledge and the cues provided by the interface are sufficient to construct the goal structure necessary to generate the action sequence required to perform a task.

2.4.2. Generating goals for actions

Prompts, button labels and menu items interact with the user's background knowledge and existing goals to create more explicit goals, which define components of the complete task and establish goals to perform specific actions. For example, a user with a goal of "play my phone messages" may see the prompt "press P to play messages" and form the goal of "Press the 'P' button."

Notice that the lowest-level goal is to execute some physical action. It is the execution of this action that will achieve the subgoal, and it is sequences of these actions that ultimately allow the user to achieve the higher-level goals. In using the model to analyse the interaction of a user with an interface, we use the term "action" to designate either (i) "atomic" actions, such as pressing a single key or clicking the mouse button, or (ii) well-practised sequences of atomic actions, which a user in the proposed user group could be expected to execute without difficulty. These sequences include such things as "Select 'Print' from the 'File' menu" for an experienced Macintosh user.

2.4.3. Interpreting feedback

When an action has been taken, the goal structure must be revised. Feedback from the system plays a critical role in this process. Goals associated with successfully completed actions must be deactivated. New goals must be added to the structure that specify new subtasks and the next correct action. The user must interpret the system's response to determine whether some current goal has been accomplished, and hence should be deactivated, or whether some progress has been made towards

a current subtask goal. A user may post a goal to undertake some kind of error recovery action if there is no indication that progress has been made. The information contained in the system response, including prompts, is used to generate new goals for additional subtasks and further actions necessary to complete these subtasks.

2.4.4. The "and-then" goal structures

In many tasks and in interactive dialogs, goals are not posted individually but as part of a structure that represents a goal and an associated sequence of subgoals that must be accomplished in a fixed order. A good example comes from a typical initial dialog that a user has with an ATM (automated teller machine). The ATM prompts the user to enter a personal identication number followed by pressing the ENTER key. The goal is something like "give the system my personal identification number", the first subgoal is "type in my personal identification number," and the second subgoal is "press the ENTER key."

We call this goal structure an "and-then" structure because it indicates that the original goal is to be achieved by accomplishing, in order, the first subgoal *and then* the second subgoal. Although the figure shows only two subgoals, there may be more, describing an ordered sequence of requirements. In the construction–integration framework the processing of these ordered, related goals is governed by flows of activation. The "and-then" construction is represented by a pattern of associative links that are intended to direct activation to the second subgoal only when the original goal is still active and the first subgoal has been completed, as shown in Figure 1.

Each goal or subgoal shown in Figure 1 is a pair of nodes. One node, called the "want-to" node, represents the intention to accomplish the goal. The other node represents the achievement of the goal and is called the "done-it" node. Activation of a "done-it" node deactivates the associated "want-to" node. When the main goal is activated, activation flows to the first subgoal, and actions (or subsidiary goals) to achieve that goal are initiated. The "done-it" node associated with the first subgoal is activated when feedback from the environment and from internal expectations indicate that the subtask is complete. Activation from the first subgoal's "done-it" also flows to the "and-then" node, permitting activation to flow from the original goal to the second subgoal, which in turn causes the actions associated with the second subgoal to be executed.

This goal structure can account for a common error in executing a sequence of actions. If the first subgoal is similar to the original goal, the "done-it" nodes for both the original goal and the first subgoal goal are likely to be associatively connected by the propositions that represent accomplishment of these two closely related goals. These connections can cause premature deactivation of the original goal when only the first subgoal has been accomplished. We call this the *supergoal kill-off* phenomenon; Young *et al.* (1989) call it the *omitting secondary subgoals* problem; and it is sometimes called the *semicolon* problem because it occurs in coding statements in programming languages that are supposed to end with a semicolon.

In the ATM dialog example presented earlier, the original goal, "give the personal identification number" is very similar to the first subgoal, "type in personal identification number." This similarity may cause the "done-it" nodes for both goals

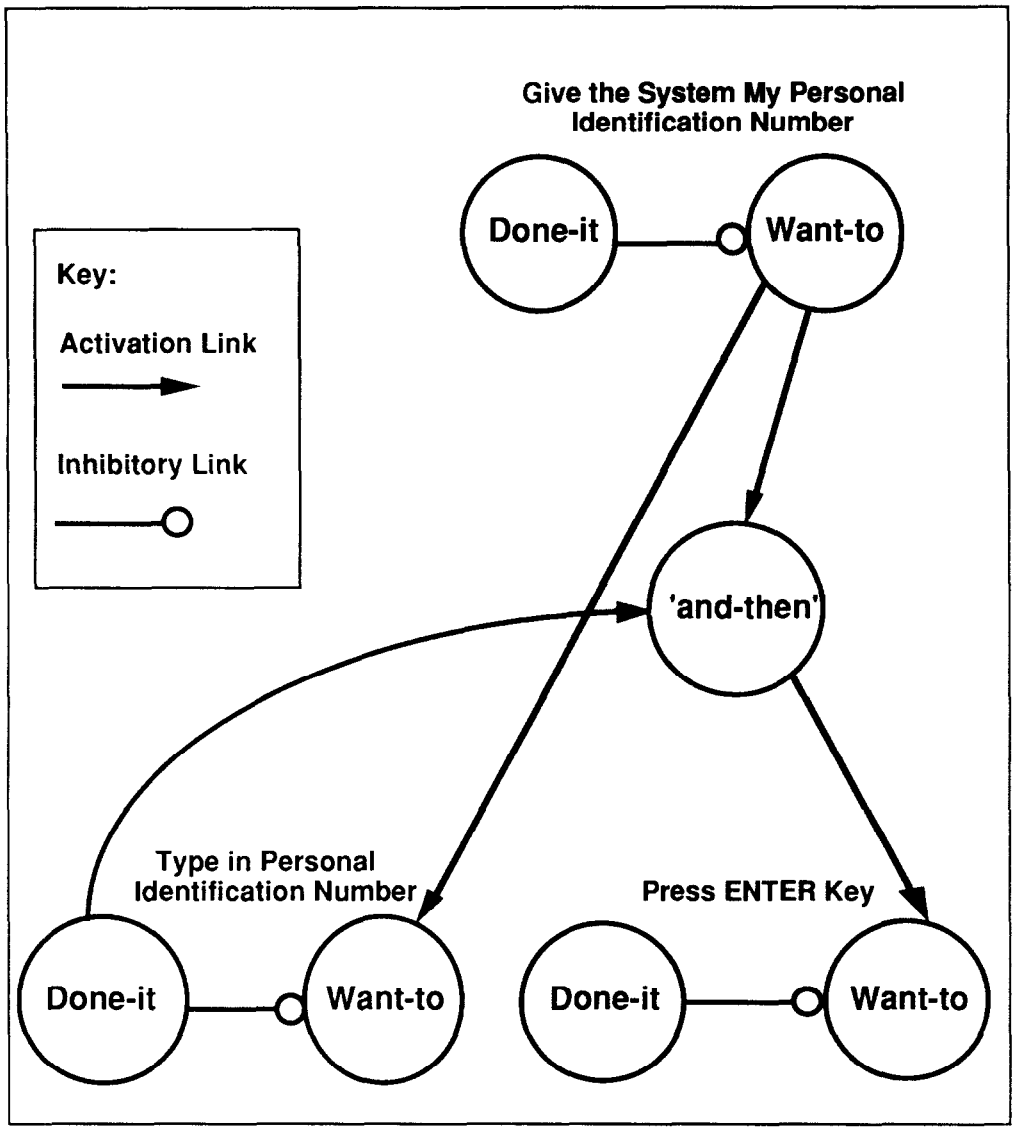

FIGURE 1. The "and-then" goal structure for the task of entering your personal identification number, broken down into the two subtasks of typing in the number followed by pressing the enter key.

to become active when the first subgoal is accomplished. This will cut off activation to both the original goal and the first subgoal. Cutting off activation to the original goal will prevent activation from reaching the node representing the second subgoal, since both inputs to the "and-then" node have to be active in order for activation to flow to the second subgoal. The terminator, the ENTER key in our example, will not be pressed because of this premature loss of activation to the whole "and-then" goal structure.

The cognitive walkthrough procedure includes a check for possible supergoal

kill-off. Whenever a subgoal in an "and-then" structure is satisfied, the original goal is checked for similarity. As will be seen in the example detailed in Section 4, "and-then" structures are common, and often include further "and-then" structures nested below the subgoals.

3. The cognitive walkthrough procedure

The cognitive walkthrough is a precisely specified procedure for simulating a user's cognitive processes as the user interacts with an interface in an effort to accomplish a specific task. A printed form with specific questions guides the walkthrough. These questions reflect the cognitive model described above, allowing an analyst without extensive training in cognitive psychology to implicitly use the model to analyze an interface. The approach is similar to that of an income tax form, which allows a lay person to evaluate the tax effects of transactions without specialized training in accounting and tax law.

A cognitive walkthrough has two phases, preparation and evaluation. In the preparation phase, the evaluators select a suite of tasks that are a representative sample of the tasks that the application is intended to support. For each task in the suite, they describe the initial state of the interface, the action sequence used to accomplish the task and the user's initial goals. During the evaluation phase, the interaction between the user and the interface is analysed in depth. In accordance with the cyclical model underlying the theory, the analysts look at each user action to determine (a) what goals the user should have leading up to the action, (b) whether the prompts and labels of the interface will induce the user to take the correct action, assuming the correct goals and (c) how the user's goals will change in response to the feedback from the interface after the action is performed. The steps of the walkthrough are laid out on paper forms shown in Figures 2, 3a–c. Terminology used in the walkthrough, which was discussed in the previous section, is summarized in Table 1.

3.1. PHASE 1: PREPARATION

3.1.1. Choose tasks to analyse

The first step in preparing for a walkthrough is to select a suite of tasks to be analysed. These should be tasks whose accomplishment the system is intended to support, defined and delimited from the point of view of the user. How tasks should be selected is outside the scope of the method, though the value of the results clearly depends on choosing a suite of tasks that represent things users actually will want to do. The suite should include tasks that are made up of sequences of elementary or basic tasks. In our experience, transitions among subtasks are often sources of problems for users.

For example, a bank-by-phone application could have "query account balances," "transfer funds from one account to another," and "query to see if specific checks have cleared" as basic tasks supported by the application. The user will rarely have trouble with these basic tasks in a reasonably designed system. However, many tasks will involve sequences of these basic tasks. Consider the task of preventing the last check you wrote at the supermarket from bouncing. You first see if the check

TABLE 1
Terminology used in the cognitive walkthrough

task	An activity that a user would want to perform with the system being analyzed. Examples: • "Check the spelling of file 'foo'" • "Log in to the computer."
goal	Something the user wants to achieve. Higher-level goals may be identical to task descriptions, while lower-level goals will describe actions. Examples: • CHECK SPELLING OF FILE "FOO" • START WORD PROCESSSING PROGRAM • PRESS THE "ENTER" KEY
action	A physical activity that the user can perform. This may be a very simple, "atomic" action, or a well-practiced sequence of atomic actions. Examples: • Press the 'S' key • Select the "File" from the "Print" menu
goal structure	A hierarchy of related goals. Under each higher-level goal, the subgoals may be sequenced ("and-then") or unordered. Examples: • LOG ON TO THE COMPUTER ENTER LOGIN NAME and-then ENTER PASSWORD • DELETES FILES 'FOO' and 'BAR' DELETE 'FOO' DELETE 'BAR'
and-then	A goal structure in which two or more subgoals must be performed in a specified order. (See example above.)
step	The unit of analysis in the cognitive walkthrough procedure. At each step the analysts consider three things related to a single action: • What goals the user should have immediately before the action; • Whether these goals, given the current state of the interface, will cause the user to select and execute the correct action; • How the interface's response to the correct action will cause the user's goals to change.

has cleared. If not, you query your checking account balance. If the balance is too small, you transfer the necessary funds from your savings account. Decomposing the original task into the right collection of subtasks and then executing the subtasks in the correct order is a much more severe test of the interface.

3.1.2. Provide a task description

Once the tasks have been chosen, the walkthrough form guides the rest of the preparation phase. The task must be described from the point of view of the first-time user. Any special assumptions about the state of the system when the user begins work must be noted. It is important to describe the tasks in realistic terms, without using system-specific terms that real users would not use. For more complex tasks, this initial description will often involve listing the primary subtasks, i.e. the task decomposition, that must be accomplished by the user in order to complete the original task. Background knowledge required to generate the correct task decomposition is also recorded on the form.

Cognitive Walkthrough Start-up Sheet

Interface _____
Task _____
Evaluator(s) _____ **Date** _____

Task Description: Describe the task from the point of view of the first-time user. Include any special assumptions about the state of the system assumed when the user begins work.

Action Sequence: Make a numbered list of the atomic actions that the user should perform to accomplish the task.

Anticipated Users: Briefly describe the class of users who will use this system. Note what experience they are expected to have with systems similar to this one, or with earlier versions of this system.

User's Initial Goals: List the goals the user is *likely to form* when starting the task. If there are other likely goal structures list them, and estimate for each what percentage of users are likely to have them.

FIGURE 2. A simplified version of the form used to record the results of the first phase of the walkthrough. The evaluators record the task description, the initial state of the interface (both visible and hidden state information), and the "correct action" sequence. (The actual forms allow more space for the analysts to write in than is shown in the figures in this article.)

3.1.3. Determine correct sequence of actions

For each task to be analysed, an appropriate sequence of actions must be specified for accomplishing the task using the system. The cognitive walkthrough analysis will critique this action sequence. It is important that the sequence be the best offered by the interface; that is, it should be the sequence that would be expected to offer the fewest obstacles to users. If an inferior sequence is used in the analysis and problems are found, little has been learned. However, if the best sequence available is shown to have problems, then there are clear grounds for modifying the system. Who specifies the action sequence may vary, but the designer of the system may be best suited to define the path which the interface was designed to support. If the system is intended to offer alternative acceptable sequences for a task, all should be analysed.

Choosing the right level of granularity for the action sequence requires some judgement. If each keystroke is treated as a separate action, the sequence may be at to low a level, be long and require much trivial and repetitive analysis, e.g. treating each keystroke as a separate action when the user types in a long, familiar word. On the other hand, if actions are clumped together, the analysis will be done at too high a level, and some potential problems will be missed. The supergoal kill-off problem discussed earler will be missed if the subgoals "type in personal identification number" and "press ENTER key" are collapsed into one action, "enter personal identification number." The best policy is to start by thinking of actions being very finely divided, then to collapse them into larger units only when it is clear that no potential problem will be hidden.

If the interface is still in the design stage, the designer will need to provide detailed specifications of the interface's appearance before each action, and of the interface's behavior when each action is executed. Annotated sketches of screens are appropriate for a visual interface. For an audio interface, proposed messages should be quoted. Timing of responses is also relevant information that the analysts will need to know.

3.1.4. Identify anticipated user population

Throughout the walkthrough procedure, the analysts will be required to make detailed predictions about the users of the interface: what goals they will they form, what actions they will find easy or hard, what changes to the interface they will recognize as progress. To establish a sound footing for these judgements, a relatively uniform population of potential users needs to be identified before the analysis is started. The form asks the analysts to specifically consider the user group's background knowledge that might be relevant to the interface. This could include experience with different applications on the same system, with similar applications on other systems, or perhaps with similar interfaces. The knowledge assumed for the user group will be a powerful tool during the rest of the walkthrough, justifying the analysts' conclusions and helping to focus questions that could otherwise be difficult to resolve.

3.1.5. Describe the user's initial goals

The final item on the form in Figure 2 marks the transition between the preparation phase and the evaluation of the interface. Here the analyst notes what goals the user will have at the start of the interaction. The goals listed should be those the user is likely to form when starting the task. Each goal should be annotated as to whether it is based on the task description, cues present initially on the screen or background knowledge. If the goals include "and-then" structures, the goal and all subgoals for each structure should be shown, with the leading, initially active subgoal marked. If there are other likely goal structures, they should be listed, and estimates should be given as to what percentages of users are likely to have them.

The point here is to indicate what the user's goals *are actually likely to be* at this stage, rather than what they should be for the task to be accomplished. When the walkthrough begins these probable goals will be compared with the goals needed to perform the task with the system, and discrepancies will count as problems with the interface. Kieras (1988) presents a method for generating goal structures.

3.2. PHASE 2: EVALUATION OF EACH ACTION

During the evaluation phase, the interaction between the user and the interface is analysed in depth. A complete analysis, described below, is performed for each required action identified in the preparation phase and listed on the form shown in Figure 2. Figures 3*a*, *b* and *c* show different parts of the form used to guide and record the analysis of each action; as many copies of the form are filled in as there are actions in the sequence.

The items on the form check for possible failures of the exploration process model outlined earlier. Most items also ask the analyst to estimate the percentage of users

for whom a particular step in the exploration process will succeed; these percentages can be used to sort out which failures are the most serious.

The form is divided into three sections. The first section deals with the relationship between the goals needed to operate the interface and the goals users actually have (Figure 3*a*). The second section checks for problems in selecting an action, given the appropriate goals (Figure 3*b*). The third section of the form asks the analyst to work out how users' goals are likely to change after making the correct action and observing the system's response (Figure 3*c*).

3.2.1. Goal structure for the current step

User problems with an interface can be divided into *goal* problems and *action* problems. In a goal problem, the user is trying to do the wrong thing; in an action problem, the user is trying to do the right thing but cannot figure out how to do it. To permit problems to be divided in this way, the walkthrough procedure asks the analyst to specify, at each step, what the correct goal(s) for this step are. As with the sequence of correct actions, the correct goals might be supplied by the designer, reflecting his or her view of how users should approach the task. Also, like the action sequence, these goals should be chosen to make the interface look as good as possible.

By specifying the appropriate goals differently, designers (or analysts) can change the way problems look in the walkthrough analysis and what the obvious remedies for them would be. For example, suppose the right action at some step of an interaction is "press the 5 key," and suppose that the user's goal at this stage is likely to be "end this transaction." If the designer indicates that "end this transaction" is the appropriate goal here, there will be obviously be an action problem. There is no apparent way the user would link the 5 key to the current goal. Possible remedies would include providing a label linking the 5 key to the goal, providing a key labelled "end" and using it instead of the 5 key and so on. If, on the other hand, the designer specifies that the appropriate goal for this step is "press the

Cognitive Walkthrough For A Step

Task _____ **Action#**_____

1.Goal structure for this step.

1.1. Correct goals. What are the appropriate goals for this point in the interaction? Describe as for initial goals.

1.2. Mismatch with likely goals. What percentage of users will not have these goals, based on the analysis at the end of the previous step? Check each goal in this structure against your analysis at the end of the previous step. Based on that analysis, will all users have the goal at this point, or may some users have dropped it or failed to form it? Also check the analysis at the end of the previous step to see if there are unwanted goals, not appropriate for this step, that will be formed or retained by some users. (% 0 25 50 75 100)

FIGURE 3*a*. A version of the form used to record the results of the second phase of the walkthrough for each user action. This figure shows Section 1 out of three major sections.

5 key," there would be no action problem but there would be a goal problem: users probably won't have that goal. A natural remedy here would be to provide a prompt directing users to press 5 to end the transaction, or to train users so that they know that pressing the 5 key is needed at this step.

This flexibility can be confusing, but as the example shows, the different choices for appropriate goals reflect different choices in design philosophy which the designer can make. Generally, a designer can work with more natural goals, and try to solve the resultig action problems, or can assume less natural goals and try to solve the resulting goal problems.

Item 1.2, of Figure 3a, presents the series of questions that guide the process of evaluating the match between the goal structure that the user is likely to have and

2. Choosing and executing the action.

Correct action at this step: _____

2.1. Availability. Is it obvious that the **correct** action is a possible choice here? If not, what percentage of users might miss it? (% 0 25 50 75 100)

2.2. Label. What label or description is associated with the **correct** action?

2.3. Link of label to action. If there is a label or description associated with the **correct** action, is it obvious, and is it clearly linked with this action? If not, what percentage of users might have trouble? (% 0 25 50 75 100)

2.4. Link of label to goal. If there is a label or description associated with the **correct** action, is it obviously connected with one of the current **goals** for this step? How? If not, what percentage of users might have trouble? Assume all users have the appropriate goals listed in Section 1. (% 0 25 50 75 100)

2.5. No label. If there is no label associated with the **correct** action, how will users relate this action to a current goal? What percentage might have trouble doing so? (% 0 25 50 75 100)

2.6. Wrong choices. Are there other actions that might seem appropriate to some current goal? If so, what are they, and what percentage of users might choose one of these? (% 0 25 50 75 100)

2.7. Time-out. If there is a time-out in the interface at this step does it allow time for the user to select the appropriate action? How many users might have trouble? (% 0 25 50 75 100)

2.8. Hard to do. Is there anything physically tricky about executing the action? If so, what percentage of users will have trouble? (% 0 25 50 75 100)

FIGURE 3b. A version of the form used to record the results of the second phase of the walkthrough for each user action. This figure shows Section 2 out of three sections.

the "correct" goal structure. Once the "correct" goals have been specified they are compared with the goals users are actually likely to have, using the analysis done as part of the evaluation of the previous action (or on the preparation sheet, in the case of the first action). The analyst checks whether the "correct" goals are likely to be present, or whether some users may not have formed them, or may have discarded them prematurely. The analyst also checks whether users are likely to bring over inappropriate goals from the previous action. Missing or inappropriate goals detected here are goal problems.

3.2.2. Choosing and performing the action

The next stage in the analysis looks for possible action problems. That is, assuming users have the appropriate goals, will they choose the correct action? Goal problems are put aside, having been dealt with in the previous stage of analysis, and the information available to guide the user's choice is examined from the point of view of a user who has the correct goals.

The form in Figure 3*b* requires the analyst to check the following points. First, the action must be available: the user must know that it is a possible action (Item 2.1). For example, some menu-based systems have keyboard actions which are not indicated on the menus. Users may not know these actions exist. Second, the action must be linked to some current subgoal (Items 2.2–2.5). This can happen because the system supplies some explicit prompt or cue that connects the action to the subgoal ("press 1 for more messages") or because of background knowledge the user can be assumed to possess ("hanging up the phone will terminate this transaction"). Third, there must be no competing, incorrect action that is also well connected to current subgoals (Item 2.6). Fourth, the user must have time to make the necessary selection: some interfaces interrupt the user if a response is not made within a specified time (Item 2.7). Finally, the action itself should be easy to execute (Item 2.8); an action would fail this last test if it required multiple simultaneous key presses or selecting a small target on a touch screen.

3.2.3. Modifications of the goal structure

The final part of the analysis considers changes in the user's goal structure caused by the user's interpretation of the system's response. Assuming the user has successfully executed the correct action, what changes to current goals will he or she make? The results of this analysis feed into the check for goal problems, mismatches between the user's actual goals and the goals necessary to generate the next correct action. Since most goal modifications are triggered by the system's response to the previous action, this last part of the analysis begins with a description of that response.

The first point checked is whether the user will see that progress has been made toward some current goal (Figure 3*c*, Item 3.1). If not, the user may create a new goal here, to quit or back up. This will usually, though not always, create a different goal problem at the next step. Only if the correct next action actually is to quit or back up would this not signal trouble for the interface.

The next point in the analysis is to identify current goals that have been accomplished and consequently should be dropped. It is important that users realize these goals are accomplished, or otherwise they will be kept activated and may

3. Modification of goal structure.
Assume the correct action has been taken. What is the system's response?

3.1. Quit or backup. Will users see that they have made progress towards some current goal? What will indicate this to them? What percentage of users will not see progress and try to quit or backup? (% 0 25 50 75 100)

3.2. Accomplished goals. List all current goals that have been accomplished. Is it obvious from the system response that each has been accomplished? If not, indicate for each how many users will not realize it is complete.

3.3. Incomplete goals that look accomplished. Are there any current goals that have not been accomplished, but might appear to have been based on the system response? What might indicate this? List any such goals and the percentage of users will think that they have actually been accomplished.

3.4. "And-then" structures. Is there an "and-then" structure, and does one of its subgoals appear to be complete? If the subgoal is similar to the supergoal, estimate how may users may prematurely terminate the "and-then" structure.

3.5. New goals in response to prompts. Does the system response contain a prompt or cue that suggests any new goal or goals? If so, describe the goals. If the prompt is unclear, indicate the percentage of users who will not form these goals.

3.6. Other new goals. Are there any other new goals that users will form given their current goals, the state of the interface, and their background knowledge? Why? If so, describe the goals, and indicate how many users will form them. NOTE that these goals may or may not be appropriate, so forming them may be bad or good.

FIGURE 3c. A version of the form used to record the results of the second phase of the walkthrough for each user action. This figure shows Section 3.

cause trouble for the next action. Item 3.2 of Figure 3c asks for a list of these accomplished goals, and asks the analyst to consider for each one whether there is adequate indication to the user that it is complete.

Item 3.3 checks for the opposite problem: goals that have not been accomplished but may look as if they are. If these goals need to be carried forward to the next step, as would usually be the case, false indications that they are complete will cause trouble.

If an accomplished goal is a sub-goal in an "and-then" structure, it needs special checking, which is specified in Item 3.4. If it is the last subgoal of an "and-then" structure, then not only it but its supergoal must be deactivated. If it is not the last subgoal, then it should be deactivated and the next subgoal made active. However, if it is similar to its supergoal trouble may occur, as discussed earlier: the supergoal may be prematurely judged accomplished, and the subsequent subgoal may not be

made active. This is the supergoal kill-off problem, which often shows up as a missing delimiter in an interaction.

Items 3.5 and 3.6 check for the creation of new goals. If the system response includes a prompt or menu the user is usually supposed to form a new goal based on the information provided. Item 3.5 asks the analyst to examine whether the prompt will be effective or whether some users may not form the expected goal.

Even without a prompt users may form new goals after executing an action. For example, if they are working through an "and-then" structure a new subgoal may just have become current. They may have background knowledge about how to accomplish the new subgoal which will lead them to form further subgoals at this point. Item 3.6 asks the analyst to identify new goals that may be formed here.

These items from Figure 3c in the "Modification of goal structure" section of the walkthrough suggest probable goal problems. Failing to form the goal suggested by a prompt is probably going to lead to trouble. However, one cannot be sure about goal problems until the correct goals for the next action are considered, at the beginning of the analysis of the next action. After all, it could be that the goal suggested by the unclear prompt actually is inappropriate for the next step, so that it is users who form the goal, not those who fail to form it, who are in trouble. This point is especially pertinent to Item 3.6, which asks for new goals not formed in response to a prompt. Here there can be no presumption as to whether or not the new goals are needed, since they are not suggested by the interface.

This section of the walkthrough concludes the analysis for a single action. The proper handling of this section, then, is as input to the determination of goal problems in the next step. The percentages of users forming or not forming certain goals should not be taken as indications of problems at this stage, but just as data to be used in identifying goal problems at the initial stage of the analysis of the next action.

4. Example: walking through "forward your calls"

In this section we describe a cognitive walkthrough of a task using a phone system. We describe the start-up activities of a group of analysts, then highlight the problems they discover as each of the required user actions is analysed.

The interface we use in the example is based closely on a real one, but its original designers might well object to the analysis we present here. We will evaluate the interface as a walk-up-and-use interface, where the user is not assumed to have any knowledge of the system prior to use, other than the knowledge an average adult would have about telephone systems. As will emerge in the analysis, it is unlikely the designers intended to meet this demanding standard, but probably presumed some level of user training before use. Nevertheless, our analysis is of value, because although training for the system was offered to users many did not take it, including one of us (C. Lewis). The analysis gives a reasonable account of his real-life experience with the system.

4.1. ACTIVITIES BEFORE THE WALKTHROUGH

The analysts begin by agreeing on a task and a sequence of actions that will achieve the task. One of the analysts is assigned to be the recorder. The recorder writes the

group's initial decisions on the walkthrough start-up sheet (Figure 2). The task is described as:

Forward all my calls to 492-1234.

The system start-up state has the special characteristic that the calls are already transferred to 492-1111. The correct action sequence is:

1. Pick up the handset.
2. Press ##7.
3. Hang up the handset.
4. Pick up the handset.
5. Press **7.
6. Press 21234.
7. Hang up the handset.

The recorder copies each of the actions onto a separate walkthrough form (into the space at the top of the form in Figure 3*b*).

To understand this sequence, the analysts examine a transparent overlay for the telephone keypad which provides instructions to users. They see that the sequence consists of two parts, clearing forwarding (pressing ##7) and then specifying forwarding (pressing **7 followed by entering the number that calls are to be forwarded to).

The target group of users have no knowledge of this system or any similar systems, and the recorder notes this on the form. The group now must indicate the user's initial goals. They feel nearly all users will have the goal included in the task description, "forward all calls to 492-1234," and that nearly all will form the "and-then" subgoals "pick up handset" and "specify forwarding" for this goal. They discuss the fact that not all users may assume they must pick up the handset before specifying forwarding, but decide the number of users who will have this problem will be small. They also consider whether users are likely to form the goal of clearing forwarding before specifying it, and conclude this is unlikely for users unfamiliar with the system. They indicate that only 25% of users will form this goal. They summarize their conclusions as follows:

75% of users will have
FORWARD ALL CALLS TO 492 1234
PICK UP HANDSET
and then SPECIFY FORWARDING

25% of users will have
FORWARD ALL CALLS TO 492 1234
PICK UP HANDSET
and then CLEAR FORWARDING
and then SPECIFY FORWARDING

They use indenting to indicate when goals are subgoals of the goals above them. That is, indented goals indicate the approach the user intends to take to accomplish the goal above. They use underlining to indicate currently active goals.

4.2. ANALYSIS OF ACTION 1: PICK UP THE HANDSET

The analysts think long and hard before specifying the correct goals for starting the task, in Item 1.1 in Figure 3*a*. They can see that they face a problem that will show

up in one of two ways. If they take the correct goals to be something close to the initial goals they have just indicated are natural, there will be serious execution problems. Users will not know they need to clear forwarding, and therefore will not choose correct actions. If they make the correct goals more complete, execution errors may be avoided but there will be serious goal problems. Users will not naturally have the goal of cancelling forwarding.

They decide to let the problem show up as a goal problem, arguing that this approach keeps them closer to the designer's probable intent. They indicate that the correct goals for this step are an "and-then" structure with the top goal of "forward calls to 492 1234" and subgoals "pick up handset, clear forwarding, and specify forwarding." They show the correct goals as follows:

FORWARD ALL CALLS TO 492 1234
PICK UP HANDSET
and then CLEAR FORWARDING
and then SPECIFY FORWARDING

Having decided on the correct goals the analysts compare these in Item 1.2 with the goals they indicated were likely on the preparation form. They indicate on the form that 75% of users would be expected to have a goal mismatch at this step, because of the "clear forwarding" sub-goal that is needed but not likely to be formed.

The analysts now move in Section 2 of the form, shown in Figure 3*b*, and consider whether users will take the correct action, "pick up handset," given that they have the correct goals. They see no problems here: the action is available, and though there is no label or description for it, none is needed, since users are assumed to have the specific goal of performing this action. Coming to Item 2.6, which asks whether there is some incorrect action that users might choose, the analysts note that some users might jump the gun and try to press **7 to forward calls or **3 to send them, but conclude few will do this.

Reaching Section 3 of the form (Figure 3*c*) the analysts assume the correct action is taken, and note that the system's response is a dial tone. They also note that all users will consider that they have made progress, since they can tell they have picked up the handset, even without hearing the dial tone. In Item 3.2 the analysts indicate that the "pick up handset" goal is complete, and that all users will recognize this. In Item 3.3 they do not note any goals that might appear complete which are not. In Item 3.4 they note that "pick up handset" is the first subgoal of an "and-then" structure, so that the next subgoal, "clear forwarding," should become active. They do not think "pick up handset" is similar to the supergoal, "forward calls to 492 1234," and so they do not think any users will wrongly drop any goals here. They do not indicate any new goals in Items 3.5 and 3.6.

4.3. ANALYSIS OF ACTION 2: PRESS #7

The analysts begin another sheet (another copy of Figures 3*a–c*) to record their analysis of the next section. They are happy with the same correct goal structure as for the first step, except that the second subgoal, "clear forwarding," is active.

They note in Item 1.1 that these correct goals match the goals users will bring forward from the previous step.

FORWARD ALL CALLS TO 492 1234
PICK UP HANDSET
and then CLEAR FORWARDING
and then SPECIFY FORWARDING

In Section 2 the analysts note that "press ##7" is an available action, and that there is descriptive material on the template that refers to it. They see that ##7 appears twice on the template, once in the block of text

CALL FORWARD ALL
START **7
CLEAR ##7

and again in the similar block below it:

CALL FORWARD BZY/DNANS
START **4
CLEAR ##7

Item 2.3 now asks them to consider whether this label material is obvious, and whether it is clearly linked to the action. The analysts discuss this and conclude that while most users are likely to see these labels, some may not because of the amount of label material to be scanned. They indicate that about 25% of users may have trouble here.

In Item 2.4 the analysts consider whether the labels for ##7 are clearly linked to the goal of clearing forwarding. Because of the word "clear" in the labels they are happy about this, and they decide to let this go. They indicate 0% trouble in Item 2.4.

At Item 2.6 the analysts note a serious problem. In addition to the labels for the correct action the template includes

TRANSFER ALL CALLS
START **3
CLEAR ##3

and the analysts fear that CLEAR TRANSFER ALL CALLS is as good a match to "clear forwarding" as CLEAR CALL FORWARD ALL is. They indicate that 25% of users might choose action "press #3" at this step. The analysts see no problems in Items 2.7 and 2.8.

In Section 3 of the form the analysts note that the system's response to pressing ##7 is a series of short beeps: *bip bip bip*. They decide that while this message is not very informative to users who are unfamiliar with the system, few users would decide to quit or back up at this point.

In responding to Item 3.2, they debate whether forwarding has now been cancelled, or whether it is not cancelled until the handset is hung up in the next step. By experiment they determine that forwarding really is cancelled before hanging up. So the subgoal "clear forwarding" is complete at this point, but because of the

uninformative feedback not many users will be confident of this. However, the analysts decide that although many users will be unsure of what has happened most will proceed on the assumption that forwarding has been cancelled rather than keeping this goal open and looking for another way to clear. In responding to Item 3.3, they do not see any goals that are incomplete but might appear complete.

Since "clear forwarding" appears in an "and-then" structure the analysts consider Item 3.4. They note that the 50% of users who realized that forwarding has been cancelled should make the final subgoal "specify forwarding" active. They do not judge that "clear forwarding" is similar enough to the supergoal to cause trouble. They do not note any new subgoals in Items 3.5 and 3.6.

4.4. ANALYSIS OF ACTION 3: HANG UP

The analysts begin another sheet (Figures 3a–c). In considering the correct goals for this step, they note a problem. The goals carried over from the previous step will not make the user hang up here. Rather than proceeding with these goals and noting an execution problem, they decide they will go back and add a new subgoal to the "and-then" structure they indicated at the first step of the task. The new structure has the supergoal "forward calls to 492-1234" and subgoals "pick up handset, clear forwarding, hang up, and specify forwarding." They revise their discussion of goal mismatch at the start of the task to reflect the additional requirement that users must know to hang up the handset after cancelling forwarding, and their analysis of the "and-then" structure in Item 3.4 of the previous action. Having made these changes they see no goal mismatches in Item 1.1.

FORWARD ALL CALLS TO 492 1234
PICK UP HANDSET
and then CLEAR FORWARDING
and then HANG UP
and then SPECIFY FORWARDING

The remainder of the evaluation for this action produces no surprises. In responding to the questions in Section 2, the analysts see no difficulty in carrying out the action of hanging up, and in Section 3 note that the "hang up" goal will be accomplished and the "specify forwarding" subgoal, which comes next in the "and-then" structure, will become active. In Item 3.6 they indicate that users will form the new goal of picking up the handset, as a subgoal of the newly active "specify forwarding" goal, given that the handset is now back in the cradle and users can be assumed to know that they must pick up the handset to work with it, as was assumed in formulating the initial goals for this task.

4.5. ANALYSIS OF ACTION 4: PICK UP THE HANDSET

The analysts show the correct goals at this step to be:

FORWARD ALL CALLS TO 492 1234
PICK UP HANDSET
and then CLEAR FORWARDING
and then HANG UP
and then CLEAR FORWARDING

and then SPECIFY FORWARDING
PICK UP HANDSET
and then SPECIFY FORWARDING

which agrees with the goals they expect to see carried over from the previous step. In Sections 2 and 3 they see no problems: the user will pick up the handset and will update the goal structure so that "specify forwarding" is the only current goal.

4.6. ANALYSIS OF ACTION 5: PRESS **7

The analysts indicate "specify forwarding" as the appropriate goal for this stage in the interaction, which matches the goal carried over from the last step.

The analysis for choosing the appropriate action given this goal is similar to the analysis for "press ##7." The potential for confusion among CALL FORWARD ALL, TRANSFER ALL CALLS and CALL FORWARD BZY/DNANS is again noted in Item 2.6.

Assuming the user performs the correct action the analysts decide that users will not be tempted to quit or back out, despite the absence of feedback. They also decide that users will not prematurely drop the "specify forwarding" goal because they have not specified the destination for forwarding. So the analysts decide that users can be expected to leave this step with just the "specify forwarding" goal current.

FORWARD ALL CALLS TO 492 1234
PICK UP HANDSET
and then CLEAR FORWARDING

and then HANG UP
and then SPECIFY FORWARDING
PICK UP HANDSET
and then SPECIFY FORWARDING

4.7. ANALYSIS OF ACTION 6: PRESS 21234

The analysts note two difficulties in formulating the appropriate goals for this step. First, there is no explicit indication that the destination number is to be dialed at this point, so it appears users must already have a goal posted to do this now. Second, they note that what must be dialed is not the complete number for the destination but only that part which would be dialed to reach the number from this phone. They conclude that the appropriate goal structure is:

FORWARD ALL CALLS TO 492 1234
PICK UP HANDSET
and then CLEAR FORWARDING
and then HANG UP
and then SPECIFY FORWARDING
PICK UP HANDSET
and then SPECIFY FORWARDING
and then DIAL EXTENSION FOR DESTINATION

and note the mismatch: this goal will not be carried over from the earlier step. After discussion, they conclude that some proportion of users might actually guess this goal in the aftermath of the previous step, and add an indication in Item 3.6 of the previous step that 25% of users might add the correct goal there. This leaves them with an estimate of 75% of users not having the needed goal at this step.

Given the correct goal, the analysts see no problem in executing the correct action. The system response is the same *bip bip bip* sound heard after cancelling forwarding, and the analysts note that this does not provide any clear feedback to the uninitiated user, but they nevertheless conclude that users will consider their "specify forwarding" goal to be complete. In Item 3.6 they indicate that given ordinary knowledge of the telephone, users will now form the goal of hanging up.

4.8. ANALYSIS OF ACTION 7: HANG UP

The correct goal for this step seems clear to the analysts, and it matches the goal carried over from the previous step.

FORWARD ALL CALLS TO 492 1234
PICK UP HANDSET
and then CLEAR FORWARDING
and then HANG UP
and then SPECIFY FORWARDING
PICK UP HANDSET
and then DIAL EXTENSION FOR DESTINATION
and then HANG UP

They see no problem in executing the appropriate action, and no reason to form any new goals associated with the task.

4.9. AFTER THE WALKTHROUGH

In a real design situation, the walkthrough results would now go to the designer (who may have also been part of the walkthrough team). Clearly the main problem with the interface centers on the need to clear forwarding before specifying a new destination. The designer can pursue one of three remedies: providing training so that users will bring the right goal structure to the task, supplying prompt information that tells users of the need to clear, or changing the system so that clearing is no longer required. The last solution is obviously the best, if it is technically feasible.

As problems are corrected, the designer would scan through the walkthrough forms, note the goals and prompts that had caused problems, and evaluate the new design for the same task. The detailed information recorded on the walkthrough form that identified the problem would help ensure that the new design didn't cause a different problem at the same point. However, only by reviewing the entire walkthrough could the revised interface be fully checked, since a changed label on the template might incorrectly match a goal at any point in the action sequence, and a revised goal structure might be modified inappropriately in a step that worked well for the previous goal structure.

5. Experience with the method

The most direct evidence of the effectiveness of the cognitive walkthrough procedure is presented in Lewis *et al.* (1990). The authors used an early version of the procedure to analyse four simple answering machine interfaces for which results of user testing were available. (Lewis *et al.*, 1990; p. 237, shows a copy of the walkthrough form.) The outcomes of the analyses were compared with evaluation data obtained from college students performing two elementary tasks. Results were promising but not outstanding: about half of observed user errors were identified in the walkthroughs. The false alarm rate, the proportion of errors identified in the walkthrough but not observed in user testing, was high, almost 75%.

5.1. STUDENT PROJECTS AND OTHER INFORMAL EVALUATIONS

Tests using later versions of the method in student projects by P. Arment, R. Comeaux, M. Esemplare, K. Farnes, R. Moleres, K. Rabin, C. Stewart and H. Wilcox on interfaces for a voicemail directory, a text editor, and a document routing application produced broadly comparable results. Testing with the voicemail directory showed that 50% of errors observed in thinking-aloud user tests were identified in the walkthrough. Testing with the text editor interface included about fifty varied tasks and compared the walkthrough results with a variety of evidence of user problems, including field trouble reports and user interviews. The comparison data were not matched specifically to the tasks used in the walkthrough, but nevertheless about 30% of errors identified in the walkthrough appeared in the comparison data. Of errors rated most serious in the walkthrough more than 70% appeared in the comparison data. In the most successful test, using the document routing application, the walkthrough procedure identified all problems found in subsequent testing without any false alarms. However, this test was limited to a single, fairly simple task.

The method has also been used in a number of design studies in which direct comparison data are not available. Applications have included commercial software, student projects, research tools and both telephone and graphical interfaces. Without exception participants have indicated that the method was useful in identifying potential problems in the designs.

5.2. EVALUATION OF A GRAPHICAL INTERFACE

Jeffries *et al.* (1991) evaluated the effectiveness of the cognitive walkthrough methodology in analyzing an interface that was not designed as a walk-up-and-use system. In part, this study served to actually evaluate the interface, and in part it served to compare the cognitive walkthrough methodology, as applied by a group of software engineers, to three other user interface evaluation techniques: guidelines, also applied by a group of software engineers; heuristic evaluation, applied by user-interface professionals; and usability testing of six subjects, conducted by a human-factors professional. The application evaluated was a beta-test version of a visual interface to the operating system of a general-purpose workstation. The interface provides graphical tools to manipulate files, applications, help, screen appearance etc. Seven user tasks were analyzed with the walkthrough methodology, and the same tasks were used in the usability testing.

The study compared the quantity and type of usability problems found by each of the four evaluation techniques. The heuristic evaluation by user-interface professionals identified 105 interface usability problems, while usability testing identified 31, the guidelines approach identified 35 and the cognitive walkthrough identified 35. Usability testing was not treated as a baseline method in this study, so the question of false alarms, i.e. how accurately the predictions of the other methods matched usability data, was not addressed. (See Jeffries *et al.*, 1991, for more detailed results, including evaluations of problem severity.)

The study applied measures such as severity, consistency, recurrence and generality to the usability problems found, and used these results to categorize the advantages and disadvantages of each of the four evaluation techniques. Under the criteria used in the study, heuristic evaluation by user-interface professionals was deemed to be most successful at identifying interface problems, although the authors noted that this approach required the effort of several evaluators with the knowledge and experience necessary to perform the evaluation. Usability testing was also rated high, but with several disadvantages, including high cost. The cognitive walkthrough technique, like the guidelines approach, had the advantage of being usable by software developers. It also assisted the developers in defining the user's goals and assumptions, information that could be useful in correcting problems once found. Disadvantages of the walkthrough technique included the need for a methodology to define user tasks for evaluation, the time required to apply the method, and the method's failure to identify what the study termed "general and recurring problems."

5.3. EVOLUTION OF THE METHOD

Overall, these studies suggest that persons new to the method can apply it successfully. However, people with a background in cognitive science find it easier to get started. The concepts of goal structure, such as the decomposition of a supergoal into subgoals, are new to many designers and analysts and require some explanation. Some of the tests of the method have examined differences among analysts in the problems identified. One result is that the developers of the method, we ourselves, are more successful in applying it than persons without experience with it. Esemplare, Farnes, Moleres, Rabin and Wilcox undertook attacking this problem directly, and were able to revise the walkthrough form so as to increase its effectiveness for inexperienced analysts. In general, the walkthrough form has evolved to more directly reflect the structure underlying our model of learning by exploration (see Section 2) and has become much more detailed. Compare Lewis *et al.* (1990: p. 257) with Figures 2 and 3a–c.

Besides difference in knowledge and experience, there are other factors that produce differences among analysts. Some differences are traceable to poor control of the cognitive walkthrough procedure itself, with analysts examining differing solution paths or forgetting steps in a path. Similarly, analysts make different assumptions about the goals associated with steps in the procedure, not so much because of real differences in analysis but because of differences in the care taken to completely specify the goal structure.

Further variability arises from uncertainty about facts not derivable from the cognitive theory underlying the walkthrough, such as whether or not users will know

what a "pound sign" is on a telephone keypad. Finally, there are differences in noticing: one analyst may notice a potential link between a goal and the prompt for an inappropriate action and another may not. These differences between analysts have important implications for the method. The Lewis *et al.* (1990) study and other less formal evaluations have focused on questions concerning interrater reliability, and thus the analyses were done by individuals. We have done a few analyses as a group and find it a very different experience. Even experienced individuals are sensitive to different aspects of an interface. The group discussion provides a means for evaluating and integrating these different viewpoints.

Another conclusion from all experiences with the method is that it is tedious. Filling in the forms is repetitious and requires a lot of writing. We are exploring the possibility of an automatic prompting system, which (for example) would permit correct goals to be specified by revising those specified for the previous step. We think this would not only speed up the procedure considerably but would also eliminate much of the variability among analysts noted above.

5.4. CURRENT APPRAISAL

Our current appraisal of the method is favorable. We, and others who have tried the method, feel that it provides a systematic way to produce an analysis of an interface in considerable depth with much less effort than is required by explicit modelling approaches. While not all problems in an interface can be identified, the cognitive walkthrough analysis provides a framework in which empirical questions, like the one about the "pound sign" used as an example above, can be identified and their impact on the design understood, so that empirical testing of the interface can be more focused.

6. Comparison with other evaluation methods

In this section, we review other design methods that have been developed to improve usability. The cognitive walkthrough is a competitor, or a complement, to these other methods.

6.1. THEORETICALLY-BASED SIMULATIONS AND ENGINEERING MODELS

6.1.1. GOMS/CCT

The best known and controversial proposals for applying cognitive theory to the design process are to use theoretically-based, simulation models to evaluate trial designs. The most developed of these proposals are the GOMs model (Card, Moran & Newell, 1983) and Cognitive Complexity Theory (CCT) (Kieras & Polson, 1985; Polson, 1987). These models range in complexity from simple engineering models that can be developed in a few minutes to a few hours (e.g. the keystroke model of Card, Moran & Newell, 1983), to complete, running, cognitive simulations (Bovair, Kieras & Polson, 1990). These models all deal with routine cognitive skills and are capable of making quantitative predictions of performance time and, in the case of CCT, training time.

The use of these models can supplement, but not replace cognitive walkthrough analysis. The models do not include the processes involved in exploration and they

do not attempt to predict errors driven by either slips or conceptual confusions (Norman, 1981). Their use, accordingly, does not reveal the kind of problems the cognitive walkthrough aims to identify.

6.1.2. PUMs

Programmable user models, or PUMs, (Young, Green & Simon, 1989; Young & Whittington, 1990) are similar to CCT in requiring that the analyst write an executable specification of the mental operations involved in using a system and the knowledge about the task and system possessed by the new user. However, unlike CCT, PUMs are based on SOAR (Laird, Newell & Rosenbloom, 1987), a problem-solving architecture that models the acquisition of cognitive skills. PUMs can predict some of the conceptual problems new users will have during their attempts to learn the new system.

To use a PUM the analyst constructs a set of "instructions" which can be interpreted by a model of the user's problem-solving behavior. The instructions supply the knowledge necessary to guide the problem-solving model to use the interface to be analysed. Problems with the interface appear whenever it proves difficult to find instructions that work.

6.1.3. Comparisons with cognitive walkthroughs

There is much in common between the PUM method and the cognitive walkthrough. Both methods aim to expose the way in which users' mental processes can be expected to succeed or fail in attempting some task with an interface. However, there are important differences as well. First, work with PUMs has focused on situations in which users know quite a bit about the operations available in a system, so that fairly complex reasoning is possible. The cognitive walkthrough focuses on situations in which users know little about the system, and hence must be guided by superficial cues rather than relying on their own reasoning. If either method were to be extended toward the situation addressed by the other, they would probably converge.

A related difference is that the cognitive walkthrough focuses on *the interface being analysed,* with questions designed to relate difficulties in mental processing to specific features of the interface, while PUM analysis focuses on *the user's knowledge and reasoning.* In situations in which users know a good deal about the system, and can reason effectively about it, the cognitive walkthrough orientation toward relatively superficial aspects of an interface might suffer in comparison with the PUM's orientation.

Another difference, at least considering the thrust of early PUM work, is that the cognitive walkthrough directs the analyst to think through the sequence of mental operations needed to perform a task, using this analysis to suggest what knowledge may be needed to support these operations, while PUM analysis seeems to focus on identifying needed knowledge *a priori.* Thus the walkthrough asks *where problems are likely to occur* in attempting a task, while the PUM analysis asks *what instructions will suffice* to support the task. The cognitive walkthrough analyst directly constructs a critique of the interface, by identifying likely problems, while the PUM analyst first attempts to create an adequate collection of instructions and then critiques them, or notes the difficulties in creating an adequate set. We think

the cognitive walkthrough approach provides clearer direction for the analyst. We also do not think the approach is really inconsistent with the PUM idea. One can imagine attacking the problem of instructing a PUM starting in the same way as for a cognitive walkthrough, by determining what mental operations are apparently required in a task and building up instructions incrementally as needed to produce the needed operations.

A final important difference between the cognitive walkthrough approach and the PUM approach is the role of simulation. In PUM analysis the instructions for an interface are evaluated by running them in an executable simulation, whereas the walkthrough analysis stops with the analyst's judgement that some step will or will not be carried out. While the use of a running simulation has potential benefits in guarding against mistakes of judgement, it has serious costs as well, of three kinds.

6.1.4. Problems using executable simulations

First, considerable experience with CCT, which uses executable simulations simpler than those needed in PUM analysis, indicates that writing the simulations is an unacceptable burden (Karat, Fowler & Gravelle, 1987; Butler *et al.*, 1989). Simulation modelling requires the use of novel software tools, e.g. production systems or LISP, and other skills that are not generally available within most software design teams. Not surprisingly, designers or analysts *will not* write a large program, which can approach in complexity the system under development, just to evaluate usability. In other words, developers view proposals to develop detailed cognitive models as having an unacceptable cost-benefit tradeoff.

Second, it can be difficult to accommodate in a really executable simulation sub-processes which cannot fully be implemented. In the cognitive walkthrough, analysts make judgments about whether some text on the screen will be understood to be related to a particular goal. Reasonable, though not conclusive, opinions can be ventured on such questions, even though an executable simulation of the process of comprehending text is far beyond reach.

Third, and finally, it can be difficult to relate problems in the simulation to problems in the interface being analyzed. In the cognitive walkthrough method, the questions on the form represent the results of mapping failures in mental processing back onto the aspects of interfaces and tasks that create them. The questions identify problems that would appear if a simulation were to be run, in terms of the circumstances that cause them. When working with a real simulation, this mapping back to the interface requires careful, detailed analyses of the simulation output.

6.2. OTHER WALKTHROUGHS

Bias (1988), Molich and Nielsen (1990) and Nielsen and Molich (1990) have proposed other ways to adapt walkthrough methodology to evaluate user interfaces. In Usability Walkthroughs (Bias, 1988) a panel of representative user and system designers step through tasks with a proposed system design to detect and analyse any confusions that might arise. Nielsen and his colleagues' proposals are less structured. A group of users, designers or other individuals is given a brief lecture on a short list of principles of usability and then carries out an action-by-action walkthrough of a proposed design.

Nielsen and Molich (1990) present some interesting evaluation results for their method. They identified likely problems in three system designs, and then asked evaluators to critique the designs. Individual evaluators detected a minority of the errors identified by the authors. However, different evaluators found different problems, and combining the judgments of three to five evaluators yields a large percentage of the errors with reasonable false alarm rates.

These alternative walkthrough methodologies differ from the cognitive walkthrough in two related ways. First, they do not ask evaluators to consider systematically the mental operations in system use, such as goal formation. Second, they are not tied to any theoretical model of user–system interaction. Both of these differences may favor the cognitive walkthrough approach. The usability principles in Nielsen and Molich (1990) do not appear adequate to detect the most serious problems in the telephone forwarding example described earlier, which hinge on users' goals and how they are modified. If more complete models of mental processes are developed, they can be incorporated in the cognitive walkthrough framework. However, there is no good way to update these other walkthrough procedures to incorporate new insights.

The cognitive walkthrough has already evolved in response to extensions to the underlying theory. The walkthroughs reported in Lewis *et al.* (1990) did not include consideration of "and-then" structures, and hence could not detect supergoal kill-off problems. The current walkthrough form handles this matter, reflecting growth in the theory in the meantime. We doubt that untrained analysts would detect this problem in the Nielsen and Molich method.

This is not to say that less-structured walkthroughs are of no value, or even that they will perform less well in practice than the cognitive walkthrough. Many problems with current user interfaces are so glaring that they would be revealed under any serious scrutiny. Any user test would catch the problem with cancelling forwarding in the telephone example, and it may be that most evaluators would catch it even if the Nielsen and Molich principles do not point to it. Comparative tests are required to establish the added value of the cognitive walkthrough method.

6.3. GUIDELINES

The most commonly used methodology is the application of design guidelines for usable systems. The most ambitious compendium of these guidelines is the volume by Smith and Mosier (1986); other examples are included in an introductory book on software human factors written by Rubenstein and Hersch (1984). These guidelines range from explicit pieces of advice about screen format to general statements about design methodology and the characteristics of cognitive processes.

While guidelines can provide useful guidance in the early stages of design, there are pitfalls in relying on them. For some, there is little empirical or theoretical support. Others are distillations of experience with interface technology that is now out-of-date, e.g. 24×80-character-oriented displays. Other guidelines are clearly valid but may be difficult to apply. For example, one typical guideline is "minimize working memory load." Unfortunately, this does not specify how to measure working memory load, nor does it suggest methods for reducing it. By their nature, guidelines cannot handle complex interactions and trade-offs among design features:

the complex logic necessary to describe such phenomena cannot be captured in a simple statement.

The cognitive walkthrough method does not cover the same ground as guidelines. The cognitive walkthrough examines only whether users will be able to complete a task, not (for example) whether they will enjoy it. Thus a guideline like "avoid blaming the user for errors" is beyond the scope of the cognitive walkthrough method, though clearly useful.

On the other hand, the cognitive walkthrough can give guidance on some questions that resist treatment in simple guidelines. By analysing a prompt in the context of a particular step in a task, for example, one can see clearly what will make it work well or poorly: the links to specific goals and actions are crucial. This insight is more helpful than "use simple and natural dialog," one of the guidelines used in the Nielsen and Molich (1990) study.

6.4. ITERATIVE DESIGN USING EMPIRICAL EVALUATION METHODS

Gould and Lewis (1985) and Gould (1988) urge the use of iterative design, using user testing to evaluate successive designs and discover areas for improvement. Results using the cognitive walkthrough clearly show that user testing will find problems that the walkthrough misses, and we therefore believe user testing is essential in developing a high-quality user interface, even if the cognitive walkthrough is also used.

However, user testing is more difficult to arrange than a cognitive walkthrough, because of the need to locate and work with representative users. We believe that doing a cognitive walkthrough on a design as a preliminary to user testing would be a worthwhile investment. At least some likely problems could be identified and solved before user testing, making the testing more effective, and testing could be aimed at areas of the interface where questions arise in the walkthrough.

A further benefit of the cognitive walkthrough as a complement to user testing is that it provides an explanation for problems that it identifies, whereas problems seen in user testing are often difficult to diagnose. For example, it can be unclear from observation whether incorrect user actions reflect goal problems, where users take appropriate actions in pursuit of an inappropriate goal, or execution problems, where users have the right goal but cannot find the appropriate action for that goal. The walkthrough analysis can suggest which kind of failure is more likely.

6.5. CLAIMS EXTRACTION

Carroll and colleagues (Carroll & Campbell, 1989; Carroll & Kellogg, 1989; Carroll, 1990) have proposed that user interface design should be guided by the analysis of existing interfaces, and the extraction of lessons, or "claims," from them to be applied to new designs. Proponents of this approach argue that these lessons can substitute for cognitive theory as a support for design (Carroll, 1990).

We think the cognitive walkthrough is a useful tool for extracting claims from existing designs, and can therefore support this approach. However, rather than providing a substitute for cognitive theory, the cognitive walkthrough makes it easier to apply cognitive theory to see what makes a design work well or poorly.

6.6. SUMMARY OF CONTRASTS WITH OTHER DESIGN EVALUATION METHODS

We collect, in this section, the characteristics that distinguish the cognitive walkthrough from other methods.

Role of simulation
The cognitive walkthrough does not use an executable simulation of the user's mental processes. Rather, the analyst steps through a partial hand simulation of these processes, looking for circumstances that would cause trouble if the process were to be fully modelled.

Focus on mental operations
The cognitive walkthrough asks the analyst to consider the mental processes of users in detail, rather than examining only the characteristics of the interface being evaluated.

Use of task context
The cognitive walkthrough identifies problems only in the context of specific tasks. It does not attempt to enforce general guidelines that apply regardless of task context, and it does not ask analysts to make general judgements about good and bad features of an interface regardless of task context.

Links to the interface
The cognitive walkthrough ties the analysis of users' mental processes directly to features of the interface being evaluated, so that the role of specific prompts and cues can be assessed.

Role of theory
The cognitive walkthrough is derived from a theoretical model of mental processes in exploration, and can be updated as this theory becomes more complete or is found to be in error.

7. Other uses of the cognitive walkthrough method

We have focused on the use of the walkthrough method by a group of analysts to evaluate a user interface design for a walk-up-and-use system. We see opportunities to apply and adapt the method in other situations.

7.1. CAPTURING DESIGN RATIONALE

Many workers are pointing to the importance of design rationale, the reasons why a system is the way it is, in the course of design, in redesign and maintenance (MacLean, Young & Moran, 1989; Carroll, 1990). The cognitive walkthrough provides a framework for spelling out a design rationale for a user interface, in which a designer can specify tasks to be supported, goal structures assumed for these tasks and specifically, how the prompts and cues supplied by the interface are intended to guide the evolution of users' goals during task performance.

7.2. DESIGNERS' SELF CRITICISM

Because it requires modest effort, we think the cognitive walkthrough lends itself to use very early in design, when a designer is evaluating his or her own preliminary design ideas rather than submitting them in more finished form for review by outside analysts. The method might enable designers to get a design in better shape on their own before calling on usability specialists, whose comments coming from outside a design effort may be unwelcome.

John M. Carroll (pers. comm., 1990) and others have suggested to us that the logic behind the cognitive walkthrough could be applied directly to produce design ideas, not just to evaluate them. Items on the walkthrough form would be modified to elicit specifications of interface features that would support the user at a given step, rather than asking for an evaluation of existing features. For example, instead of asking what label is provided for an action, and whether it is linked to the action and a current goal, the modified form would ask the designer to supply a label that *is* linked to action and goal.

7.3. SYSTEMS THAT ARE NOT WALK-UP-AND-USE

We have used cognitive walkthroughs informally to analyse issues in some applications in which users are assumed to have considerable knowledge of the system, for example the table features of Microsoft Word. We have been impressed that the method seems able to cope well here, even though user background knowledge of word processors and Macintosh interface conventions plays a central role. The judgements the analysts must make in this situation may be more difficult than in the case of walk-up-and-use interfaces, since analysts must weigh whether the user's background knowledge is adequate to interpret the cues correctly. The work of Jeffries *et al.* (1991) also considered this issue to some extent.

Another issue in these more complex examples is that the intended path through an interface may include searching for information, as, for example, in exploring the pulldown menus on the menu bar. It appears that cognitive walkthrough analysis can be used to assess whether this kind of exploration should be expected to work or not for a given task.

8. Conclusions

This paper has presented a theoretically-based walkthrough methodology for the evaluation of user interfaces. Our initial efforts and the initial efforts of colleagues in othe laboratories suggest that the walkthrough methodology presented in this paper has great promise.

The authors gratefully acknowledge research support from US West Advanced Technologies and the National Science Foundation, grant number IRI 87-22792. Cathleen Wharton was supported by a graduate fellowship funded by Hewlett-Packard Laboratories. The opinions expressed in this paper are those of the authors and not necessarily those of any of the three supporting organizations.

Catherine Marshall of US West has provided ideas and direction for much of this work. We thank Pamela Arment, Catherine Ashworth, Steven Coffin, Robert Comeaux, Susan Davies, Stephanie Doane, Mary Esemplare, Kris Farnes, Roland Hübscher, Douglas Lhotka, Rick

Moleres, Karen Rabin, Teresa Roberts, Claudia Stewart and Harold Wilcox for assistance and suggestions. We also thank three anonymous reviewers for valuable comments on an earlier version of this paper.

References

BELLOTTI, V. M. E. (1990). A framework for assessing applicability of HCI techniques. *Proceeding of Interact90, 3rd IFIP Conference on Human–Computer Interaction,* Cambridge, England, August 1990.

BENNETT, J. L. (1984). Managing to meet usability requirements: establishing and meeting software development goals. In J. BENNETT, D. CASE, J. SANELIN & M. SMITH, Eds. *Visual Display Terminals.* pp. 164–184, Engelwood Cliffs, NJ: Prentice-Hall.

BENNETT, J., LORCH, D., KIERAS, D. E. & POLSON, P. G. (1987). Developing a user interface technology for use in industry. *Proceeding of Interact87, 2nd IFIP Conference on Human–Computer Interaction,* pp. 21–26, Stuttgart, September 1987.

BIAS, R. (1988). User interface walkthroughs with representative users and usability experts. Paper read at the symposium *Human Factors Methods (th)at Work,*" *Annual Meeting of the Human Factors Society,* Anaheim, CA, October 1988.

BOVAIR, S., KIERAS, K. E. & POLSON, P. G. (1990). The acquisition and performance of text editing skill: a production system analysis. *Human Computer Interaction* **5,** 1–48.

BUTLER, K., BENNETT, J., POLSON, P. & KARAT, J. (1989). Predicting the complexity of human–computer interaction: report of the workshop on analytical models. *SIGCHI Bulletin* **20,** pp. 63–79.

CARD, S. K., MORAN, T. P. & NEWELL, A. (1983). *The Psychology of Human-Computer Interaction.* Hillsdale, NJ: Erlbaum.

CARROLL, J. M. & CAMPBELL, R. (1986). Softening up hard science: reply to Newell and Card. *Human Computer Interaction* **2,** 227–250.

Carroll, J. M. & CAMPBELL, R. (1989). Artifacts as psychological theories: the case of human–computer interaction. *Behavior and Information Technology* **8,** 247–256.

CARROLL, J. M. & KELLOGG, W. A. (1989). Artifact as theory-nexus: hermeneutics meets theory-based design. In *Proceedings CHI'89 Human Factors in Computer Systems,* pp. 7–14, New York: Association for Computing Machinery.

CARROLL, J. M. (1990). Infinite detail and emulation in an ontologically minimized HCI. In *Proceedings of CHI'90 Conference on Human Factors in Computer Systems,* pp. 321–327, New York: Association for Computing Machinery.

DOANE, S., KINTSCH, W. & POLSON, P. G. (1990) *UNIX command production: what users must know.* ICS Technical Report #90-1, Institute of Cognitive Science, University of Colorado, Boulder, CO 80309-0345, USA.

ENGELBECK, G. E. (1986). Exceptions to generalizations: implications for formal models of human–computer interaction. Masters thesis, Department of Psychology, University of Colorado, Boulder, CO.

FAGAN, M. E. (1986). Advances in software inspections. *IEEE Transactions on Software Engineering* **SE-12,** pp. 744–751.

GENTNER, D. R. & GRUDIN, J. (1990). Why good engineers (sometimes) create bad interfaces. In *Proceedings of CHI'90 Conference on Human Factors in Computer Systems,* pp. 277–282, New York: Association for Computing Machinery.

GOULD, J. D. (1988). How to design usable systems. In M.Helander, Ed. *The Handbook of Human–Computer Interaction,* pp. 757–789. Amsterdam: North-Holland.

GOULD, J. D. & LEWIS, C. H. (1985). Designing for usability—key principles and what designers think. *Communications of the ACM* **28,** 300–311.

JEFFRIES, R., MILLER, J. R., WHARTON, C. & UYEDA, K. M. (1991.) User interface evaluation in the real world: a comparison of four techniques. In *Proceedings of the CHI'91 Conference on Human Factors in Computer Systems,* pp. 119–124. New York: Association for Computing Machinery.

KARAT, J., FOWLER, R. & GRAVELLE, M. (1987). Evaluating user interface complexity. In

Proceeding of Interact87, 2nd IFIP Conference on Human–Computer Interaction, pp. 489–495, Stuttgart, September 1987.

KIERAS, D. E. (1988). Towards a practical GOMS model methodology for user interface design. In M. Helander, Ed. *The Handbook of Human–Computer Interaction.* pp. 135–157, Amsterdam, NV: North-Holland.

KIERAS, D. E. & POLSON, P. G. (1985). An approach to the formal analysis of user complexity. *International Journal of Man–Machine Studies* **22,** 365–394.

KINTSCH, W. (1988). The role of knowledge in discourse comprehension: a construction-integration model. *Psychological Review* **95,** 163–182.

LAIRD, J., NEWELL, A. & ROSENBLOOM, P. (1987). SOAR: An architecture for general intelligence. *Artificial Intelligence* **33,** 1–64.

LEWIS, C. H., POLSON, P. G., WHARTON, C. & RIEMAN, J. (1990). In *Proceedings of CHI'90 Conference on Human Factors in Computer Systems,* pp. 235–241, New York: Association for Computing Machinery.

MACLEAN, A., YOUNG, R. M. & MORAN, T. P. (1989). Design rationale: the argument behind the artifact. In *Proceedings CHI'89 Human Factors in Computer Systems,* pp. 247–252, New York: Association for Computing Machinery.

MANNES, S. M. & KINTSCH, W. (1991). Routine computing tasks: planning as understanding. *Cognitive Science* **15,** 305–342.

MOLICH, R. & NIELSEN, J. (1990). Improving a human–computer dialogue: what designers know about traditional interface design. *Communications of the ACM* **33,** 338–348.

NIELSEN, J. & MOLICH, R. (1990). Heuristic evaluation of user interfaces. In *Proceedings of CHI'90 Conference on Human Factors in Computer Systems,* pp. 249–256, New York: Association for Computing Machinery.

NORMAN, D. A. (1981). Categorization of action slips. *Psychological Review* **88,** 1–15.

NORMAN, D. A. (1986). Cognitive Engineering. In D. A. NORMAN & S. W. DRAPER, Eds. *User Centered Systems Design: New Perspectives in Human–Computer Interaction,* pp. 31–61, Hillsdale, NJ: Lawrence Erlbaum Assoc.

NORMAN, D. A. (1988). *The Psychology of Everyday Things.* New York: Basic Books.

POLSON, P. G. (1987). A quantitative theory of human–computer interaction. In J. M. CARROLL, Ed. *Interfacing Thought: Cognitive Aspects of Human–Computer Interaction,* pp. 185–235, Cambridge, MA: Bradford Books/MIT Press.

POLSON, P. G. & LEWIS, C. H. (1990). Theory-based design for easily learned interfaces. *Human-Computer Interaction* **5,** 191–220.

RUBENSTEIN, R. & HERSH, H. M. (1984). *The Human Factor: Designing Computer Systems for People.* Burlington, MA: Digital Press.

SHACKEL, B. (1984). Designing for people in the information age. In B. SHACKEL, Ed. *Human–Computer Interaction–INTERACT'84,* pp. 9–18, Amsterdam: North-Holland.

SMITH, S. L. & MOSIER, J. N. (1986). *Guidelines for designing the user interface software.* Report 7 MTR-10090, Esd-Tr-86-278. Bedford, MA: Mitre Corporation.

WHITESIDE, J., BENNETT, J. & HOLTZBLAT, K. (1988). Usability engineering: our experience and evolution. In M. HELANDER, Ed. *The Handbook of Human–Computer Interaction,* pp. 791–817, Amsterdam: North-Holland.

WINOGRAD, T. & FLORES, F. (1986). *Understanding Computers and Cognition.* Norwood, NJ: Ablex Publishing Corp.

YOUNG, R. M., BARNARD, P., SIMON, T. & WHITTINGTON, J. (1989). How would your favorite user model cope with these scenarios? *ACM SIGCHI Bulletin* **20,** 51–55.

YOUNG, R. M., GREEN, T. R. G. & SIMON, T. (1989). Programmable user models for predictive evaluation of interface designs. In *Proceedings of CHI'89 Conference on Human Factors in Computer Systems,* pp. 15–19, New York: Association for Computing Machinery.

YOUNG, R. M. & WHITTINGTON, J. (1990). Using a knowledge analysis to predict conceptual errors in text-editor usage. In *Proceedings of CHI'90 Conference on Human Factors in Computer Systems,* pp. 91–97, New YorK: Association for Computing Machinery.

YOURDON, E. (1989). *Structural Walkthroughs,* 4th edn. Englewood Cliffs, NJ: Yourdon Press.