The Streamlined Cognitive Walkthrough Method, Working Around Social Constraints Encountered in a Software Development Company

Rick Spencer
Microsoft Corporation
One Microsoft Way
Redmond, WA 98052-6399 USA
(425) 936-1138
ricksp@microsoft.com

ABSTRACT
The cognitive walkthrough method described by Wharton et al. may be difficult to apply in a large software development company because of social constraints that exist in such companies. Managers, developers, and other team members are pressured for time, tend to lapse into lengthy design discussions, and are sometimes defensive about their user-interface designs. By enforcing four ground rules, explicitly defusing defensiveness, and streamlining the cognitive walkthrough method and data collection procedures, these social constraints can be overcome, and useful, valid data can be obtained. This paper describes a modified cognitive walkthrough process that accomplishes these goals, and has been applied in a large software development company.

Keywords
Cognitive Walkthrough, Usability Inspection

INTRODUCTION
The cognitive walkthrough (CW) method was designed to evaluate the learnability of software interfaces without the overhead of full-blown empirical usability lab testing. The CW can be applied early in the design process because it can be applied when only the user interface is specified. As a result, the CW method is valuable for evaluating learnability of the integration of features when those features are at various stages of development.

I have used the cognitive walkthrough method to evaluate software at a software company under the constraints of an actual software development cycle. This paper describes the problems I encountered applying the Wharton, et al. cognitive walkthrough method (WCW) [4] within these constraints, and outlines a modified CW process that works better. The effectiveness and validity of the modified method, is examined as well.

CHI '2000 The Hague, Amsterdam

The CW method was used several times over the course of one software development cycle to evaluate user-interface specification for the latest release of an established Integrated Development Environment (IDE). An IDE allows programmers to edit code, design user interfaces, compile code, debug code, and perform most other programming tasks within a single computer program. An IDE, therefore, tends to be extremely feature rich, and requires a fairly large team, with distributed design responsibilities, to design and implement [1].

The design team for this product is large – approximately 50-100 people – and many different people have responsibility for specifying, designing, and implementing the UI. For many tasks that users will perform with the IDE, no one specification describes completely how that task will be accomplished. Rather, a set of specifications, each written and owned by a different person, must be understood and evaluated to understand how users will accomplish tasks with the IDE when it ships.

The CW method seemed quite well suited to evaluating the learnability of the IDE, since a single evaluation could tie together the work of several sub teams and the specifications for which they were responsible.

Wharton et al. proscribe performing a task analysis for the UI elements in question, and walking through the task analysis step by step with the team. For each step, the team attempts to tell a plausible story for each of four questions (See Table 1). For steps where plausible answers are generated by the team, the team records those stories, otherwise they record that a plausible story could not be told.

Table 1

4 questions from Wharton et al. (1994)

1. Will the user try to achieve the right effect?
2. Will the user notice that the correct action is available?
3. Will the user associate the correct action with the effect that user is trying to achieve?

4. If the correct action is performed, will the user see that progress is being made toward the solution of the task?

PROBLEMS APPLYING THE CW METHOD

In the first application of the CW method the cognitive walkthrough procedure detailed in Wharton et al [4] was followed. However, the method failed to produce good results and the development team did not perceive it as useful, at least in part because the WCW method did not accommodate some of the social constraints (SC) of a large software development company.

Table 2

Social Constraints That Hampered the Effectiveness of the CW Method

SC 1 – Time pressure

SC 2 – Lengthy design discussions

SC 3 – Design defensiveness

SC 1 – Time Pressure

Generally, many managers and developers feel pressure to make very good use of their time, and any activities perceived to take more time than justified by the results are avoided. When developing software, requirements, constraints, and changes are applied to a specification over time. As a result, numerous design iterations occur late in the development process, when a development team usually feels considerable pressure to actually implement specifications, and may not think they have the time to evaluate them properly.

Even user interface specialists often feel that there is not sufficient time in the software development cycle to perform their jobs well [1].

Voluminous output of WCW sessions

Following the procedure outlined in Wharton, et al., a plausible story is written down at every step where a plausible story is produced. As a result, the team writes down things like, "From previous experience, the user knows that the Print command is available in the File menu" and "From previous experience, the user knows that the Print command will activate the Print dialog."

Recording such obvious observations may not be perceived by the team as a good use of their limited time. Furthermore, what purpose such written comments will serve in the future may not be clear, and the ratio of useful written comments (problems, design ideas, and design gaps) will be diminished, diminishing the perception of usefulness for the CW exercise.

Finally, sorting the potential problems identified from the plausible stories may lengthen the time from the CW to when the problems are reported.

CW sessions seem to go too sowly

When doing a WCW the team answers four questions for each step. This results in the team dwelling on each step, even ones that are obviously designed correctly because they follow standard user-interface convention, or because that part of the user interface has been working for users over many previous versions. Furthermore, similar plausible stories are often repeated multiple times for the same step.

For steps with obvious problems, asking the four questions can seem redundant, especially if the team has had difficulty distinguishing between the first three questions and has brought up the issues pertinent to questions 2 and 3 while attempting to answer question 1.

Others have also found the CW method to be too slow. For instance, the Cognitive Jogthrough method was developed for the sole reason that one development team found that they were unable to cover enough material while using the CW method [3].

SC 2 - Lengthy design discussions

WHEN A design or user-interface problem is identified by a design team, that team will often attempt to resolve the problem "in-line" during the CW session. Time allotted to evaluation is spent designing instead.

SC 3 - Design Defensiveness

As surprising as it may seem, it turns out that not everyone who puts considerable effort into creating user interface specifications enjoys having those specifications publicly evaluated by others. Specification writers may appear personally offended by the suggestion that there specifications should undergo an evaluation process in the first place, because, after all, they may have been working on those specifications for many months, or even a year or more.

Since, in the short term, problems that are identified may result in more work for a team that could already be under considerable time pressure, some team members may try to defend their designs and specifications during the CW, may be argumentative, and may reject seemingly obvious observations as being opinions that lack data to support them.

CONDUCTING A STREAM-LINED COGNITIVE WALKTHROUGH

The three social constraints that limit the effectiveness of the WCW method have been addressed through a combination of approaches. Namely, properly preparing the team to perform a CW, modifications to the CW method itself, and strong leadership during the CW process to keep team members from dwelling on design discussions or defending their designs.

Table 3

Overview of the Cognitive Walkthrough process, adapted directly from Wharton, et al [4]

1. Define inputs to the walkthrough
    a. Identification of users
    b. Sample tasks for evaluation
    c. Action sequences for completing the tasks
    d. Description or implementation of interface
2. Convene the walkthrough
    a. Describe the goals of the walkthrough
    b. Describe what will be done during the CW
    c. Describe what will not be done during the walkthrough
    d. Explicitly defuse defensiveness
    e. Post ground rules in a visible place
    f. Assign roles
    g. Appeal for submission to leadership
3. Walkthrough the action sequences for each task
    a. Tell a credible story for these two questions:
    - Will the user know what to do at this step?
    - If the user does the right thing, will they know that they did the right thing, and are making progress towards their goal?
    b. Maintain control of the CW, enforce the ground rules
4. Record critical information
    a. Possible learnability problems
    b. Design ideas
    c. Design gaps
    d. Problems in the Task Analysis
5. Revise the interface to fix the problems

DETAILED DESCRIPTION OF THE STREAMLINED WALKTHROUGH PROCEDURE

1. Define inputs to walkthrough

Before the CW session, the usability professional is responsible for defining the important user task scenario or scenarios and producing a task analysis of those scenarios by explicating the action sequences necessary for accomplishing the tasks in the scenarios. Wharton et al. should be used as a resource for determining how to decide on the scenarios and how to describe the task sequence.

2. Convene the walkthrough

The first step is to describe the goals of performing the walkthrough. Namely, the walkthrough is an opportunity to evaluate the user interface in terms of learnability. This is the first opportunity to address SC 3 – Design defensiveness, by defusing defensiveness on the part of any team members. It is important that the usability professional points out that learnability is only one aspect of usability, and that the team recognizes that learnability may have been traded off for other aspects of usability. Nonetheless, there is inherent value in knowing when users may encounter problems learning an interface as the issue could be explicitly addressed elsewhere, for example, though marketing or the help system.

A CW session is analytical in nature, and therefore lacks the definitiveness of an empirical usability tests. In light of the CW method's tentative nature, specification owners may resent absolute proclamations that "this is a problem". The usability specialist should, therefore, take care to use softer language, like "this is a potential problem, we need to think about it". Constant reference to the tentative nature of the finding should help defuse defensiveness.

The usability professional then points out for the first time that the CW is an evaluation session, not a design session, and goes on to describe the process of walking through the task sequence and answering the two questions for each step (See table 4). The usability professional then gives an example of an action sequence from software not currently under consideration and that has plausible answers to the two questions, and the team is encouraged to produce those answers. Then the usability professional gives another example, one without plausible answers, and the team is prompted to try to provide answers. For each example, the usability professional should model the format that the data is captured in before proceeding with preparing the team for the CW.

Table 4

2 questions from the streamlined CW

1. Will the user know what to do at this step?
2. If the user does the right thing, will they know that they did the right thing, and are making progress towards their goal?

After describing what the team will do during the walkthrough, describe what the team will *not* do during the walkthrough. This is the first opportunity to directly address SC 2 – Lengthy design discussions, and indirectly address SC 3 – Design defensiveness. In particular, the usability professional explains that if the team finds a step with possible learnability issues, they will note the possible problem and move on to the next step, but they won't redesign the interface. Furthermore, the usability professional should explain that if the team encounters a gap in the design (for example when it is not clear from the specification what action sequence the user is supposed to perform), the team will note the gap and move on, but they won't stop and design the missing actions. Also, if a design idea is suggested, the team may briefly discuss the design idea, note it, and then move on, but the team will not flesh it out. Lastly, if the task analysis appears to be faulty, or only describes one of multiple possible paths towards achieving

the goal, then the problem will be noted and the team will move on, but the task analysis will not be revised during the CW session.

After the team understands what will and will not be done during the walkthrough session, the usability engineer explicitly addresses SC 3 – Design defensiveness. Since the CW session is an evaluation session, and not a design session, no changes will be made during the CW session. If changes are going to be made to specifications, they will be made later. Therefore, if anyone feels that a specification needs to be defended, the time to do so is later. More importantly, defending designs or specifications during the CW is not productive and will distract the team from completing the evaluation.

After the CW process is explained and defensiveness defused, the usability professional should post the ground rules for conducting the walkthrough. These ground rules can then be referred to later in order to keep the team on track.

Table 5

Ground rules for conducting a streamlined CW

1. No designing
2. No defending a design
3. No debating cognitive theory
4. The usability specialist is the leader of the session

Ground rule 3 is a further effort to address SC 3 – Design defensiveness. In my experience, team members who feel that their designs are threatened may appeal to esoteric cognitive theories in order to justify their designs or to explain away a possible issue. The usability professional should state that as long as a significant number of team members feel there is a possible issue, it should be noted

Responsibility for collecting data from the walkthrough is distributed across the team participating in the walkthrough. Four kinds of data will be collected, design ideas, design gaps, potential learnability problems, and flaws in the task analysis. If the team participating in the CW is large enough, and the scope of the tasks under scrutiny cover multiple areas of user interface design, then different team members can be assigned the role of collecting potential learnability problems for different areas. Generally, team members can be assigned to collecting data on the areas for which they are responsible. Usually, one person is sufficient for collecting both design ideas and design gaps; however, if there are enough team members, then one person can collect design ideas and one person can collect design gaps.

It is important to explicitly assign a role for collecting flaws in the task analysis. This helps address SC 3 – Design defensiveness, by modeling a willingness to admit to and take responsibility for oversights and mistakes.

Lastly, the usability professional should explain that in order for the CW to proceed efficiently, the team must follow ground rule 4 and submit to the usability professional's leadership. Then an explicit appeal to submit to leadership should be made.

3. Walkthrough the action sequences for each task

The proposed modified CW severely prunes the evaluation procedure for each action sequence. For each action sequence, the usability professional first describes the action sequence and the system state after the correct action is performed. Then the team tries to answer the two questions for each action sequence.

Wharton et al.'s questions 1-3 evaluate whether the user will know what the next appropriate step is, and how to do it. For the streamlined CW, these three questions have been collapsed into one question, "Can you tell a credible story that the user will know what to do?" Question 4 is slightly recast, "If the user does the step correctly, and <describe system response>, is there a credible story to explain that they knew they did the right thing?

During the course of walking through the action sequences, the usability professional must take care to enforce the ground rules, making sure that the CW session does not lapse into a design session, that team members do not stop the process to defend their designs, and that the team does not get wrapped up in debates over cognitive issues.

4. Record critical information

If, for a particular action sequence, a plausible story is told for both questions, then nothing is recorded, and the team moves on to the next task. This helps address SC 1 – Time pressure, by spending minimal time on steps that appear to be properly designed.

Sometimes a plausible story cannot be told because the interface design assumes knowledge that users might not have. In such cases, record the failure and the knowledge that the user will need to formulate the goal, for instance, "Users might not click the ellipsis button to modify the list, because they might not know that they can modify the list."

Often, the team will not try to tell a plausible story, but instead will skip right to pointing out a design flaw. In this case, a bullet should be captured that represents the problem for the user and the design flaw. For example, "Users might not know that they can modify the auto generated code because it looks the same as read only code."

During a CW session, team members will likely bring design ideas that address problems encountered or suggest totally alternative designs. It is important to capture these design ideas because they may prove valuable later, and summarizing a design idea as a bullet is a good way to end design discussions and move on. For example, a design idea

bullet may read "DI: Automatically generate the code for the user, instead of having the user issue the command." This also allows team members who raise design ideas to feel that their contribution was valued, while allowing the team to continue with the evaluation.

The usability professional is responsible for stopping a design discussion before it gets out of hand. When exactly to intercede may be affected by many factors. Generally, a good time to intercede and move on is as soon as a design idea is well-formed enough to be expressed in a bullet, but before other team members begin pointing out flaws in the design idea, or elaborating on it.

During the course of the CW session, the team may find that they forgot to design some important functionality. "How does the user do such and such a setting?" Such design gaps should be captured as a bullet, but definitely not designed during the CW session. If that gap is encountered again during the CW, the team must agree to hand wave over it, and assume that the ultimate design will support the functionality,

When a bullet is being captured, it is helpful for the usability professional to rephrase it in the form of a hypothesis. This allows the team to express consensus on the bullet that is being captured, as well as giving the person capturing that bullet a head start on writing it down properly.

If the usability professional made a mistake in the task analysis used in the CW, the problems in the analysis should be noted, and the team should move on. It is important to not try to retool the task analysis during the CW session. It is far better either skip the part that is wrong and cover it during a later session, or if the mistake was substantial, the usability professional should apologize for the misunderstanding, and reconvene the CW at a later time, after the task analysis is done properly.

Attempting to retool a task analysis in a few minutes is not likely to lead to a quality analysis. Major mistakes should be rare if the task analysis is checked for accuracy by the specification owners before the CW session.

IMPACT ON THE EFFECTIVENESS AND VALIDITY OF THE CW METHOD

Preparing the team, clearly laying out ground rules, and defusing defensiveness are steps that can be confidently added to the CW method without too much fear of decreasing the effectiveness or validity of the CW method. But what about the more radical changes, such as collapsing three questions into one question, disallowing design discussions, and capturing less data during the walkthrough? I will discuss these in turn, in order of probable severity in negative impact on the method.

Collapsing three questions into one

This modification to the CW method probably leads to a coarser-grained evaluation of the user interface under scrutiny. Logically, asking only half as many questions about an action sequence will probably lead to fewer problems identified for each step, presumably by both a function of time, less time is spent on each question, and a function of detail, each action sequence is examined in less detail.

Furthermore, when problems are encountered, the cause of the problem may not be revealed as effectively as in the WCW method. For example, the result of the first question from the streamlined CW might read, "Users might not know that the Print command will bring up the print dialog." However, the same datum from the WCW method might read, "Users might not associate the Print command with activating the Print Dialog, because the word 'Print' implies an action." The datum from the WCW seems to imply a cause and solution to the problem, where the streamlined method merely identifies the problem.

In practice, however, I have not found that the WCW does not lead to better data because the team is generally interested only in identifying problems and getting enough information to fix them. Furthermore, many people have trouble understanding the nuances of Wharton, et al.'s four questions [2].

Disallowing Design discussions

Given the best of all possible worlds, letting design discussion play out at the time that potential learnability problems are identified could lead to some very effective design sessions. When design discussions are blocked from the evaluation session, the usability professional is trading off identifying possible solutions to identified problems for coverage.

However, open-ended design sessions take an indeterminate amount of time, and may or may not result in a workable redesign. Worse stills, if the team redesigns as they proceed through the CW, then problems will be resolved in the order that they occur in the task sequence, with no necessary relationship to the severity of the problems. Time taken up by redesign may result in truncation of the CW because the team simply runs out of time. Important steps involving completing tasks may therefore be missed. In other words, redesigning during the CW violates a basic dictum of software design, to profile *before* you optimize.

However, this bias against design discussions during CW sessions is not universally shared. In fact, Rowley and Rhoades specifically created the Cognitive Jogthrough method because they found that a WCW did not allow *enough* time for design discussions [3]!

Capturing Less Data

Using the streamlined method, the plausible stories are not captured. Though this will clearly speed the process, it means that design rationale is not captured. Later in the process, if there are questions about the data from the CW, it may be difficult to remember why a particular step was considered to be acceptable by the team. Furthermore, the

design rational will not be available later in the design process.

EFFECTIVENESS OF THE STREAMLINED CW

The streamlined CW method was used to evaluate the learnability of an IDE under development. Specifically, the task of performing a series of programming tasks necessary to create and deploy the basics of a one kind of computer program was evaluated. The programming task was considered to be one of the most important programming tasks for the version of the IDE under development, and it incorporated functionality described in multiple specifications. At the time of the CW, the necessary functionality was months from being sufficiently well implemented for traditional usability lab testing.

Methods

The task analysis took one usability professional with a background in task analysis about 25 hours to complete. Since the specifications for the task were distributed across many documents, and not all of the documents were up to date, most of the 25 hours was spent researching specifications and interviewing program managers responsible for designing various parts of the user interface.

A team of approximately 8 people was convened to conduct the CW, including 3 usability specialists, 1 graphic designer, and 4 project managers responsible for various aspects of the user-interface specification. Only one usability professional leading the session was familiar with CW methods. The walkthrough itself took about 2.5 hours, and was conducted over 2 sessions, separated in time by about a week. Only the results from the first session, which took about 1.5 hours and covered 32 action sequences, are discussed here. About 20 minutes of the first session were used to prepare the team, assign roles, and defuse defensiveness. Since the task analysis spanned many specifications, most team members were not familiar with all of the action sequences. Therefore, many minutes were required to explain the user action and the system response for each of the 32 action sequences covered.

Results

Twenty-four potential problems and 11 design ideas were identified during the first 1.5-hour session. Of the 24 potential problems generated during the CW, 14 suggested that users would lack sufficient knowledge to take the correct action, and 10 suggested that the IDE did not provide good feedback to the user when the correct action was taken.. Six of the 11 design ideas were specific solutions to one or more of the potential issues.

These results are consistent with the hypothesis that the streamlined CW method trades granularity for coverage. The 10 potential usability problems identified that lacked an explicit cause also did not suggest an explicit solution. In other words, for many of the identified problems, the team agreed that they were potential problems, but the cause of the problems were attributed only to a lack of knowledge on the part of users, and not necessarily to a mismatch between the users' knowledge and the user interface. Had the team considered each of the first three questions from the Wharton method, it is possible that a better understanding of the causes of the problems would have surfaced.

Generally, the efforts to defuse defensiveness on the part of the team members were successful, as the team did not spend much time defending design decisions, and an atmosphere of cooperation seemed to prevail. When program managers found themselves defending their design, they tended to remember ground rule 2, drop the discussion, and allow the CW to continue. After the CW session, the team members expressed that it was a useful exercise, and in fact many CW sessions have been conducted since.

EXTERNAL VALIDITY

The difficulty in assessing the external validity of any usability inspection method extends, of course, to the streamlined CW as well. Ideally, the results of the CW session would be evaluated against the results of an empirical usability test, where the usability specialists conducting the test were not aware of the CW results. Naturally, such a luxury is not afforded within the scope of this effort. However, an opportunity to at least roughly assess the streamlined CW method did present itself.

Methods

Usability tests on the IDE have been conducted, and one of the usability tests did include some overlap with the material covered in the CW session. While the usability specialist who conducted the usability test was present at the CW session, he was not the usability specialist who led it. Fortunately, design changes suggested from the results of the CW sessions had not yet been implemented in the area of the IDE user interface that was being tested.

Results

Nine problems were discovered in the usability test of the UI covered in the CW session. Of those 9 problems, the CW identified 6. However, for the same user-interface elements, the usability test covered more functionality than the CW session did. For example, in most of the action sequences in the CW session, the user could accept default values to be successful, but in the user test participants needed to adjust those defaults to be successful.

This comparison suggests that, a team can expect to uncover with a streamlined CW many issues that would also be uncovered in an empirical usability test.

For the portions of the user interface covered by both the CW session and the usability test, 13 potential problems were predicted by the CW sessions. Of those, 7 were directly related to findings in the usability report, 4 were indirectly related to problems in the usability report, and 2 were not related to problems in the usability report.

The result of this comparison suggests that, for those parts of the user interface covered by a streamlined CW session,

the team can expect to hit a few false positives, get a sense of which steps may cause some problems for users, and accurately predict many learnability problems.

CONCLUSION

The Wharton, et al cognitive walkthrough method does not take into account several social realities of large software companies. The method can be applied successfully if the usability specialist takes care to properly prepare the team for the walkthrough, avoids design discussions during the walkthrough, explicitly defuses defensiveness among team members, and streamlines the procedure by collapsing the first three questions into one question, and captures data selectively.

Streamlining the walkthrough may trade-off granularity for coverage, but without that trade off, program managers and developers may perceive the walkthrough as being an inefficient use of time. Performing a streamlined CW is a good way to profile a user interface for potential problem areas, identify many steps that may be problematic for users, and accurately predict many usability problems. However the method will probably result in a few false positives.

REFERENCES

1. Grudin, J. Systemic sources of suboptimal interface design in large product development organizations. *Human-Computer Interactions 6*, 2 (1991), 147-196.
2. John, B. E, & Packer, H. Learning and using the cognitive walkthrough method: A case study approach, in *Proceedings of CHI '95* (Denver CO, May 1995), ACM Press, 429-436.
3. Rowley, D. E., and Rhoades, D. G. The cognitive jogthrough: A fast-paced user interface evaluation procedure. *Proceedings of CHI '92* (May 1992), ACM Press, 389-395.
4. Wharton, C. W., Reiman, J., Lewis, C. & Polson, P. (1994) The cognitive walkthrough method: A practitioner's guide. In J. K. Nielson, & R. L. Mack (Eds.) Usability Inspection Methods. Wiley, New York, 1994.